MSXソフト集Part.3

好評発売中

定価850円

ISBN4-416-18449-2 C2055 ¥850E

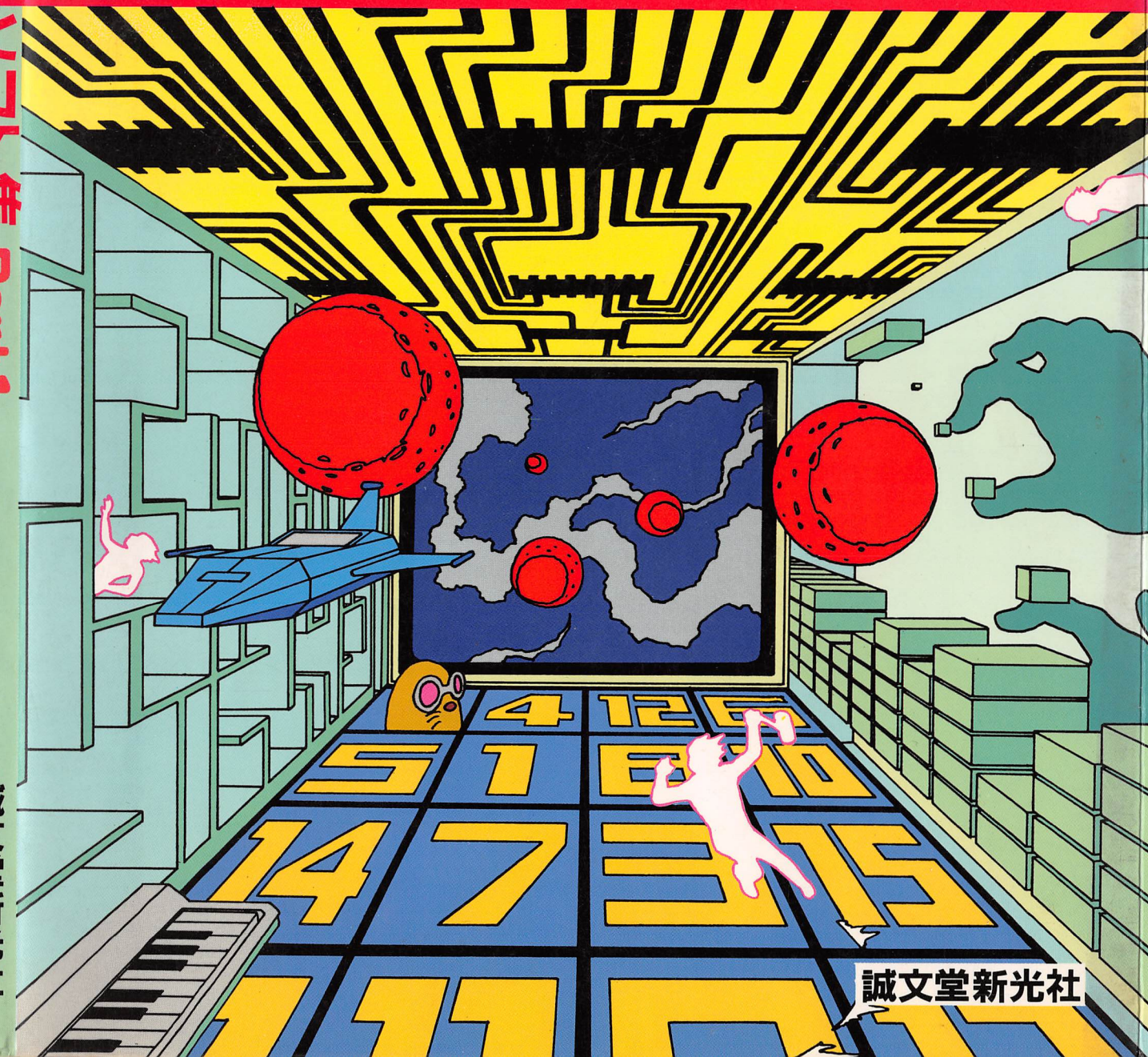
MSXソフト集 Part.1

誠文堂新光社

MSXソフト集 Part.1

誠文堂新光社

MSXソフト集 Part.2

初歩のラジオ 別冊

誠文堂新光社

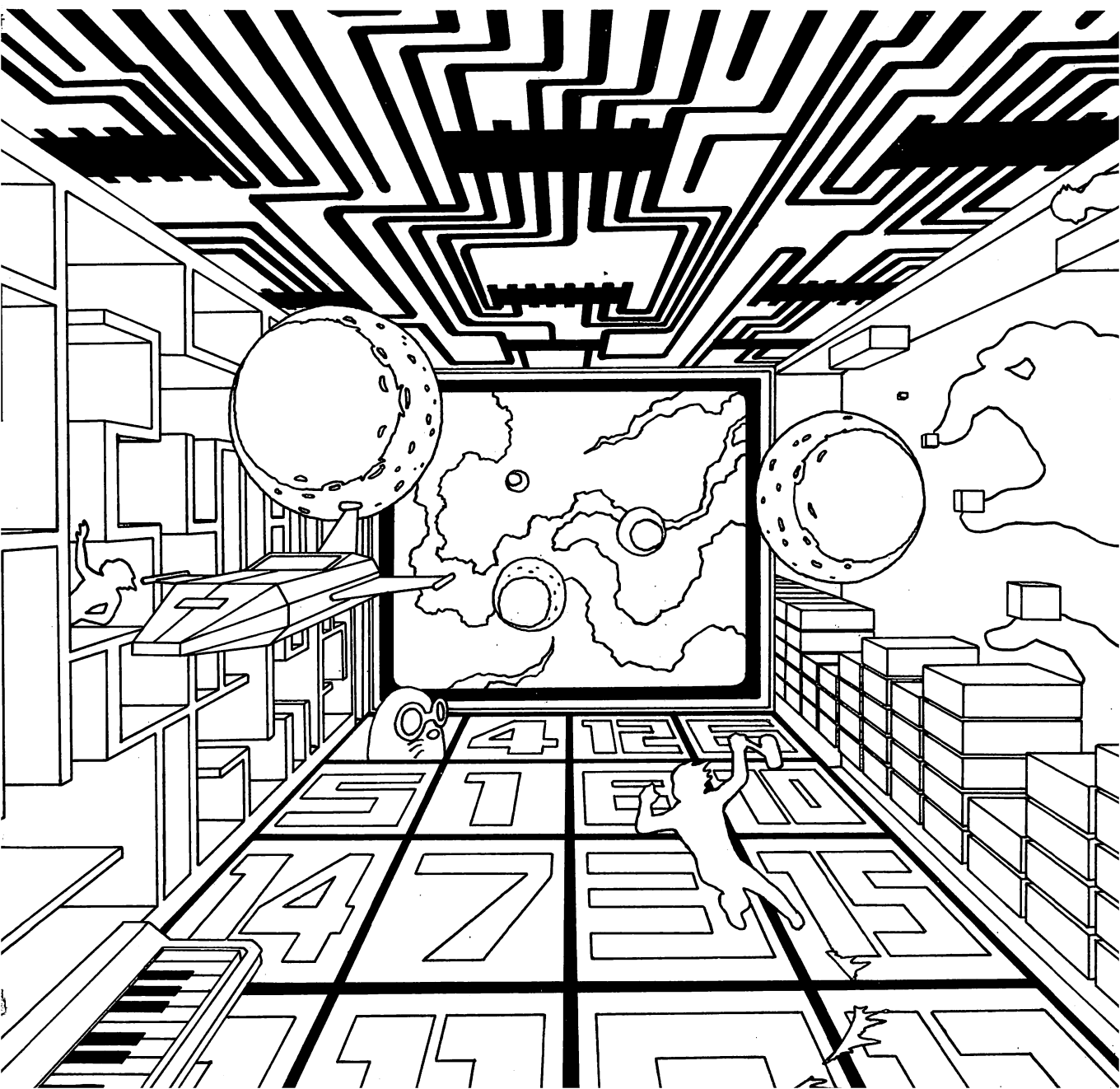
MSXソフト集Part.2

好評発売中

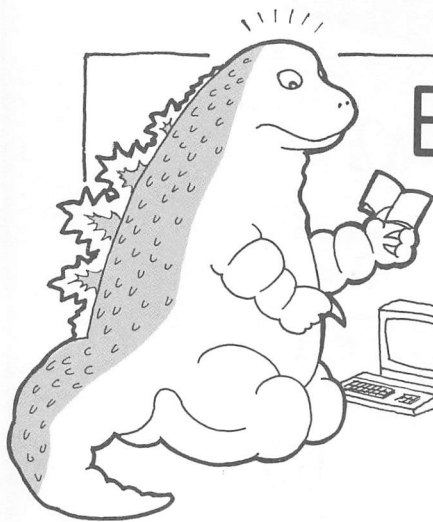
MSX ソフト集 Part.1

誠文堂新光社

MSX
ソフト集
Part.1

MSXソフト集 Part.1

MSX ソフト集 Part I

イラスト：窪野計子　表紙デザイン：JUN KIKKAWA　表紙イラストレーション：ECOV

もくじ

BASICの打ち込み方と ロード・セーブのし方

これから紹介するソフトは、すべてBASICで作っていますが、この打ち込み方とロード、セーブについて話したいと思います。

なんといってもパソコンは、プログラムがなければただのハコですね。そこでプログラムの打ち込みですが、まずリストを見てください。左はじに数字が10ごとに並んでいますが、この数字を**行番号**といいます。プログラムの入力は、この行番号ごとに横に並んだ文字を入れていくのです。

たとえば**リスト1**のようなプログラムの場合、

[1]、[0]、[スペース]、[SHIFT]+[7]、…… [RETURN]

で1行目が終りです。続いて、20、30…と入力していきますが、ひとつの行番号に始まるプログラムを打ち込んだら、かならず最後にリターンキーを押して、パソコンに登録してください。これをわすれると、たとえば20で始まる行番号のプログラムがどこで終るか、パソコンがわからなくなるわけです。

AUTO

ところで、いちいち行番号を入力するのはめんどうだと思いませんか。そんなときに便利な命令に**AUTO**というのがあります。MSXの電源を入れると、画面下に5種類の命令が表示されてますね。これはキーボードにあるF1, F2…というファンクションキーといわれるキーに対応していますが、このF2のキーに**AUTO**命令が登録されています。このF2を押してリターンキーを押せば実行されます。もちろん、自分で[A]. [U]. [T]. [O]と打ち込んでもOKで、この命令が実行されると、画面に10という数字が表示されますので、先ほど話した方法で、このあとプログラムを打ち込めば良いわけです。

リスト1

```
 10 ' Block くずし
 20 ' by Tamae Tama
 30 ' debug and advice by Haruka takagi
 40 ' copyright 1984 (C) NATS
 50 ' copyright 1984 (C)
 60 '   Seibundo-Shinkousha publishing
 70 '
 80 ' HENSU & SYSTEM INIT
 90 '
100 MAXFILES=3:CLEAR 100,&HCFFF:DEFINT A-
    000
110 DEFUSR1=&HD010:DEFUSR2=&HD021:OPEN"GR
    T AS #1
120 XS=0:YS=0:'BALL SPEED X,Y
130 XB=0:YB=0:'BALL X,Y
140 XP=0:'     PADL X
150 TP=0:'     HI- SCORE
160 SCREEN 1:CLS
```

それから、この **AUTO** 命令が働いているときにリストを全部打ち終ったらどうするかというと、いくらリターンキーを押しても、どんどん数字が出てくるだけですね。このときは、最後の行を打ち込んでリターンキーを押したら、次に[CTRL]キーと[STOP]キーをいっしょに押せば、**AUTO** 命令が解除になります。これを**ブレーク**といいます。

ところで、リストの行番号がずーっと順序よく並んでいれば良いのですが、途中で飛んでいることもありますね。たとえば510行の次が1000だったりなんてときに **AUTO** で打ち込んでいるときはどうするんじゃ、なんてことになりますね。このときは、510を打ち終り、リターンキーを押したら、ブレーク（CTRL+STOP）で一度 **AUTO** 命令を終らせて、次に **AUTO**␣1000と打ち込めば1000番地から同じように続けて打ち込めますよ。もちろんリストを全部打ち込みが終ったら、先ほどと同じようにブレーク(CTRL+STOP)で終了とします。

ところで、本書のリストは、ちょっと皆さんが見なれたものとは違うと思います。これは、リストの行番号とプログラムを見やすく分けるためにこのようにしています。もしあなたが打ち込むときに

１つの行番号が２行や３行になるときには、行番号の下のところまで右はしにいっぱいにプログラムを打ち込んでください。実際には、**リスト ３，４**のようになりますよ。

これでプログラムが打ち終ったら、こんどは**バグ取り**です。バグ取りってなんだろうって思っている人も多いと思いますが、ようするに、プログラムの打ち込みの間違いがないように確認することなのです。もちろん、実際に走らせてやる方法もありますが、まず打ち込んだリストをよーく見なければなりません。このとき使う命令に**LIST**があります。

LIST

LIST命令は、打ち込んだリストを画面に表示させる命令です。この命令も、ファンクションキーのＦ４に入っていますので、こちら

リスト２

```
820 'display all data
830 IF P=0 THEN PRINT :PRINT :PRINT "また" データ が" ありませ
    ん !! ":BEEP:PRINT :GOTO 850
840 PRINT :FOR L1=1 TO P:FOR L2=0 TO 5:PRINT USING "テ
    "ータ (#)";L2+1;:PRINT D$(L2,L1):NEXT :FOR W=0 TO 5
    00:NEXT W:PRINT :NEXT
850 PRINT "つき"の しょりを するために なにか キー を おして くた"さい !"
860 IF INKEY$="" THEN 860 ELSE 210
870 'end of job
880 PRINT :PRINT :PRINT "こ"くろうさまて"した":ON ERROR GOTO 0
    :END
890 PRINT :FOR L=0 TO 5:PRINT USING "テ"ータ (#) キーワート""
    ;L+1;:KW$(L)="":INPUT KW$(L):NEXT:RETURN
900 KA=1:FOR L1=0 TO 5:KK=INSTR(D$(L1,KS),KW$(L1)):KA
    =KA*KK:NEXT
910 IF KA=0 THEN KL=0:RETURN
920 PRINT :PRINT "***** この テ"ータ て"すか ? *****":PRINT :
    FOR L1=0 TO 5:PRINT USING "テ"ータ (#)    ";L1+1;:PRI
    NT D$(L1,KS):NEXT :KL=1:RETURN
```

リスト３

```
810 PRINT :INPUT "また" つつ"けますか ";A$:IF A$="y" OR A$="Y" THEN 780 ELSE 210
820 'display all data
830 IF P=0 THEN PRINT :PRINT :PRINT "また" テ"ータ か" ありません !! ":BEEP:PRINT :GOTO 850
840 PRINT :FOR L1=1 TO P:FOR L2=0 TO 5:PRINT USING "テ"ータ (#)";L2+1;:PRINT D$(L2,
L1):NEXT :FOR W=0 TO 500:NEXT W:PRINT :NEXT
850 PRINT "つき"の しょりを するために なにか キー を おして くた"さい !"
860 IF INKEY$="" THEN 860 ELSE 210
870 'end of job
880 PRINT :PRINT :PRINT "こ"くろうさまて"した":ON ERROR GOTO 0:END
890 PRINT :FOR L=0 TO 5:PRINT USING "テ"ータ (#) キーワート"";L+1;:KW$(L)="":INPUT KW$(L
):NEXT:RETURN
900 KA=1:FOR L1=0 TO 5:KK=INSTR(D$(L1,KS),KW$(L1)):KA=KA*KK:NEXT
910 IF KA=0 THEN KL=0:RETURN
920 PRINT :PRINT "***** この テ"ータ て"すか ? *****":PRINT :FOR L1=0 TO 5:PRINT USING "
テ"ータ (#)    ";L1+1;:PRINT D$(L1,KS):NEXT :KL=1:RETURN
930 CLS
940 PRINT "エラー か" はっせい しました!!!"
950 PRINT "もういちと" やりなおしてくた"さい"
960 INPUT "なにか キーを おしてくた"さい"
```

リスト 4

プリントしてありますが、モニター画面にはこのように出る。

```
820 'display all data
830 IF P=0 THEN PRINT :PRINT
:PRINT "まだ データ が ありません !!
":BEEP:PRINT :GOTO 850
840 PRINT :FOR L1=1 TO P:FOR
L2=0 TO 5:PRINT USING "データ (
#)";L2+1;:PRINT D$(L2,L1):NEX
T :FOR W=0 TO 500:NEXT W:PRIN
T :NEXT
850 PRINT "つぎの しょりを するために なに
か キー を おして ください !"
860 IF INKEY$="" THEN 860 ELS
E 210
870 'end of job
```

を押す方が早くて良いでしょう。

この命令を実行すると、打ち込んだリストが、次から次へと画面に表示され、とても確認するなんてむりですが、このときは STOP キーを押してください。その場で画面が一時停止になりますので、このときにリストを確認します。

もしリストに打ち込み間違いがあったら、ブレークして(CTRL+STOP キーで止める) からカーソルキーで間違ったところにカーソルを移動させて、正しいリストを打ち込みます。もしブレークさせないでカーソルを動かそうとしても動きませんし、画面にカーソルも表示されてませんよ。

LLIST

間違いがなければ、再びSTOP キーを押せば、リストの続きが表示されますので、同じ方法で確認してください。もしプリンターやプロッターを持っていれば、**LLIST** 命令を入力すれば、リストが印字されますので、それで確認してください。

確認が終ったらすぐにRUN させたいところですが、まだまだあせってはいけません。打ち込んだリストを、まずカセットなりディスクに記録しておきましょう。これは、BASIC で書かれたプログラムでも、マシン語をDATA 文として使って、実際に動くときはマシン語で動くプログラムもありますので、SAVE しておきます。このことを**バックアップを取る**といいます。

マシン語で動いているパソコンは、一度RUN させてしまうと先

ほどまで使えたブレーク（CTRL+STOPキー）が使えなくなります。ようするに、ROMカートリッジのソフトを使っているのと同じなのです。こうなると、リセットボタンを押すか、電源スイッチを切らないと止まらなくなります。ということは、間違いなく打ち込んだプログラムもその場でパーと消えちゃうのです。

また、ちゃんとプログラムが走ればまだ良いのですが、間違いがあるとまったくウンもスンもいってくれず、何もできないこともあります。このときもリセットや電源のお世話にならなければならないので、プログラムはパーですね。ですから、かならず打ち込みと確認が終ったらセーブしておく必要があるのです。

というわけで、続いて記録する命令に話をすすめます。

SAVE
CSAVE
BSAVE

このプログラムを記録しておく命令には、**SAVE, CSAVE, BSAVE**と3種類ありますが、**BSAVE**という命令はマシン語のプログラムの場合に使いますので、今回のBASICのプログラムでは必要ありません。

この**SAVE, CSAVE**は、どちらを使っても良いのですが、読み出す方法も、記録方法により決まります。

まずカセットテープに記録する場合のポピュラーな方法は、**CSAVE**命令です。使い方は、

```
CSAVE "broker"
```

という形で、最後にリターンキーを押せば、記録が始まります。この〝(ダブルコーテーション）で囲まれた6文字はファイルネームと

CSAVE"block"	SAVE"CAS:block"
OK	OK
CLOAD?	LOAD"CAS:block"
FOUND:block	FOUND:block
OK	OK
CLOAD"block"	LOAD"CAS:block",r
FOUND:block	FOUND:block
OK	

いって、プログラムの名前が入ります。この**ファイルネーム**は6文字以内でつけます。

CLOAD?

記録が終ったら、テープを録音のスタート地点まで巻き戻し、再生にし、**CLOAD**?命令を打ち込んでリターンキーを押してください。これは、プログラムがちゃんとテープに記録されたかを確認するための命令で、ベリファイとも呼ばれるものです。実行すると、

Found "(ファイルネーム)"

と表示され、最後にOKが出れば完全に記録されています。

CLOAD

読み出し方法は、もちろん**CLOAD**です。

ところでもうひとつの記録方法の**SAVE**命令ですが、これはディスクを使うときにも同じ形で使いますが、この命令をカセットテープに使う場合は、次のような形になります。

SAVE "CAS : broker"

もしCAS : を入れないと、ディスクに記録するときの命令になってしまいます。もちろんリターンキーも押さないとだめです。このときの読み出しは、

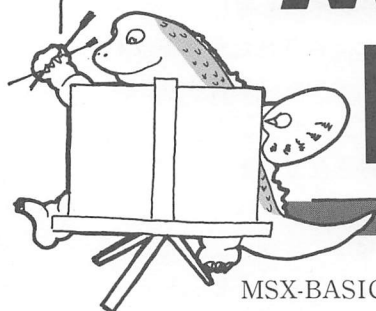
LOAD "CAS : broker"

となり、最後の" でくくられた後に，Rとすると、読み込みが終ったらすぐスタートするという命令になります。

この**CSAVE**と**SAVE "CAS :** は、同じカセットテープに記録するのにどんな違いがあるのかと思いませんか。これは、同じカセットに記録するのでも、記録される信号の形が異なるのです。**CSAVE**は、パソコンがすぐわかる形のバイナリーと呼ばれる形式で、**SAVE**は、打ち込んだリストと同じ形で記録されるアスキーと呼ばれる形なのです。2つをくらべると、同じプログラムを記録するにしても**CSAVE**の方が短くできるのですが、**SAVE**を使うと、後からプログラム同士のドッキングができる（**MARGE**命令を使う）のです。

MARGE

このような違いがあるわけですが、それぞれメリットを持っていますが、皆さんが実際に使われるときは、**CSAVE**の方が便利でしょう。

以上がセーブ、ロードのやり方です。これさえ覚えていれば、本書のゲームは君のもの。さあ、打ち込んで楽しんでください。

MSX-BASICでゲームなどのプログラムを作るときにぜひ知っておかなければならない、いくつかの特長があるので解説しておきましょう。

MSX-BASICの3大特長は、

①グラフィック（特にスプライト機能とDRAW命令）

②PSGによるサウンド機能

③豊富な割り込み機能

の3つと言えます。この3つ以外は、NECのPCシリーズや、富士通のFMシリーズなどに使用されているマイクロソフトBASICとほとんど同じですので、この3つの機能さえきちんと理解していれば、他の機種用に開発されたBASICプログラムの移植も楽になると思います。

次に、この3つの命令について詳しく解説しましょう。

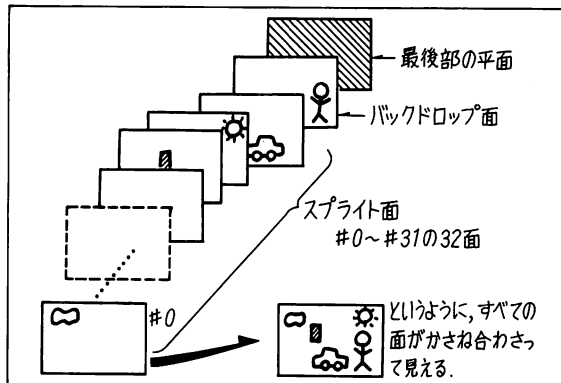
1. グラフィック機能

MSXの強力なグラフィック機能は、テキサス社の強力なLSI、**TMS9918A**によって可能になっています。このLSIは、別名VDP（ビデオ・ディスプレイ・プロセッサー）と呼ばれ、テレビゲームなどによく使用されているものです。

VDPの持つ強力な機能としてスプライト機能がありますが、まずこれについてお話しましょう。

①スプライト

スプライトは、図1のように32面をかさね合わせることが可能なパターンで、ちょうどアニメーションのセル画に相当するものです。このスプライトは、アニメーションでセル画を変えて行くことにより動きを作るのと同じく、あるスプライト面に表示されるスプライトパターンを動かすことによって、動きのあるグラフィックをいと

図1

も簡単に実現することが可能です。それゆえ、スプライトは動画とも呼ばれているわけです。

それでは、スプライトに関するMSX-BASICの命令にはどんなものがあるのでしょうか？　これを表1にまとめてみました。多くないのですぐ覚えられますね。

SCREEN
SPRITE$
PUT SPRITE

実際のプログラムでは、まず**SCREEN**命令によってスプライトの大きさを決め、次に**SPRITE$**変数にスプライト・パターンの定義を行い、**PUT SPRITE**命令で表示を行えば良いわけです。注意点としては、**SCREEN**命令を実行すると、それまで定義していたスプライト・パターンの内容が消えてしまう点と、スプライトはテキストモードでは使用できない点、それに表示する座標をうまく画面に入るようにしないとスプライトが見えなくなってしまう点です。

さらに、同一水平方向に5つ以上のスプライトがありますと、4つ目以上は表示されないので注意してください。これについては、**TMS9918A**自体がそのような仕様になっていますので、どうしようもありません。

表 I

スプライトに関する命令

SCREEN 一, 二, 三, 四, 五

二で示されるパラメーターがスプライトの大きさを示す

０：８×８ドット標準サイズ
１：８×８ドット拡大サイズ
２：１６×１６ドット標準サイズ
３：１６×１６ドット拡大サイズ

但し、スプライトは、テキスト・モードでは使用できません

PUT SPRITE 一、二、三、四

一はスプライト面の番号を示します。
この値は第一図の＃０～＃３１に対応しており、０から３１までの値をとります。

二はＳＴＥＰ（Ｘ，Ｙ）または（Ｘ，Ｙ）の表現をとるもので、スプライトの表示される場所を示しています。
ＳＴＥＰが付けられたときは、その座標は、前の座標との相対座標となります。

三はカラーを示しており、０から１５までの値をとります。

四はスプライト番号を示しており、どのスプライト・パターンを使用するかを決めます
このスプライト・パターンはあらかじめ、ＳＰＲＩＴＥ＄変数に定義されていなければなりません

SPRITE$

ＳＰＲＩＴＥ＄変数は、スプライト・パターンの定義を行うための特殊な変数で、
ＳＰＲＩＴＥ＄（１）＝ｃｈｒ＄（１）＋ｃｈｒ＄（２）＋・・・・
のように使います。
カッコの中の数字がスプライト・パターンの番号を示しており、８×８モードのときは１から２５５まで、１６×１６モードのときは、１から６３までとなっています。

スプライト・パターンのデータは、以下のように作ります

●●●○○●●●　１１１００１１１
○○●○○○○●　００１０００Ｏ１
●●●●●●●●　１１１１１１１１
●○○●●○○●　１００１１００１
●●●●●●●●　１１１１１１１１
○○●○○●○○　００１００１００
○○●○○●○○　００１００１００
●●●○○●●●　１１１００１１１

左の●と○による図形を１と０に直します
これを２進数として見てデータを作ります
すると、このパターンを定義するには、
以下のようにすればよいわけです

ＳＰＲＩＴＥ＄（１）＝＆Ｂ１１１００１１１＋＆Ｂ００１０００Ｏ１＋＆Ｂ１１１１１１１１＋＆Ｂ１００１１００１＋＆Ｂ１１１１１１１１＋＆Ｂ００１００１０＋＆Ｂ００１００１００＋＆Ｂ１１１００１１１　となるわけです

②DRAW命令

DRAW

スプライト機能と並んで重要なグラフィック命令が、**DRAW** 命令です。この命令は一口でいえば、動点（動く点）の軌跡によっていろいろな図形を描く命令で、その描きかたからタートルグラフィックなどとも呼ばれます。**表2** に **DRAW** 命令で使用されるパラメータ（サブコマンドと呼ばれます）とその使い方を示します。上手に使うと、なかなか便利なものになります。

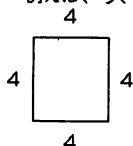
表2

DRAW命令のサブ・コマンド

Mx、y　：絶対位置x、yへ動点を移動する
M±x、±y　：現在位置からx、yだけ動点を移動する
Un　：上へnだけ動点を移動する
Dn　：下へnだけ動点を移動する
Rn　：右へnだけ動点を移動する
Ln　：左へnだけ動点を移動する
En　：右上へnだけ動点を移動する
Fn　：右下へnだけ動点を移動する
Gn　：左下へnだけ動点を移動する
Hn　：左上へnだけ動点を移動する
An　：座標系をnだけ回転する
Cn　：色を指定する
Sn　：1単位のドット数を指定する

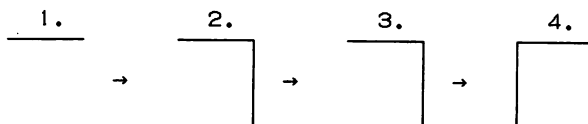
DRAW命令の使い方

例えば、今、下図のような四角形を描くことにします

4　4　4　4

まず、初点（原点）を左上とすると、右、下、左、上の順番で各4単位の線を引いて行けばよいわけです
つまり、
DRAW" R4D4L4U4"
とすれば良いわけです

この命令がどのように実行されるのかみてみましょう。
（下線の部分が現在実行している部分です）

1．右への線を引く　DRAW" R4D4L4U4"
2．下への線を引く　DRAW" R4D4L4U4"
3．左への線を引く　DRAW" R4D4L4U4"
4．上への線を引く　DRAW" R4D4L4U4"

1.　→　2.　→　3.　→　4.

2. サウンド機能

MSXにはサウンド用のLSIとして、ゼネラルインスツルメント社の**AY-3-8910**というLSIが使用されています。このLSIは別名PSG（プログラマブル・サウンド・ジェネレーター）と呼ばれているもので、マイコンの制御により多種多様な音を作り出すことが可能です。

PLAY
SOUND

MSX-BASICでは、このPSGをサポートするのに**PLAY**命令と**SOUND**命令の2つを持っていますが、初めての人は、**PLAY**命令を使いこなすことができれば十分です。表3に、**PLAY**命令のパラメーター（サブコマンド）とその使用方法を示します。このLSIは、3重和音＋ノイズ音を使用できますので、使いこなすことができれば、ゲーム作りなどがいっそう楽しくなることでしょう。

表3

ＰＬＡＹ命令のサブ・コマンド

A，B，C，D，E，F，G　：音符を示す。
音符の後に＃または＋が付くと半音高く、逆に－が付くと半音低くなる
たとえば、A＃，B＋，C－など

On	：オクターブを示す　（nは1から8まで）
Nn	：音程を示す　（nは0から96まで）
Ln	：長さ（音楽）を示す　（nは1から64まで）
Rn	：休符を示す　（nは1から64まで）
Tn	：テンポを示す　（nは32から255まで）
Vn	：音量を示す　（nは0から15まで）
Mn	：エンベロープの周波数を示す　（nは1から65535まで）
Sn	：エンベロープのパターンを示す　（nは1から15まで）
・	：符点を示す

ＰＬＡＹ命令の使い方

たとえば、ド、ミ、ソの和音を出したいときは

```
PLAY"C","E","G"
```

とする

和音でないときには、その音数だけのサブ・コマンドを送ればよい

```
PLAY"CDCDCEGFDGE","L5R2V15CDEFG"
PLAY"AB#C-DEFGDEFG"
```

などである

3.　割り込み

割り込み、という言葉を知らない人もいると思いますので、簡単に説明しておきましょう。

たとえば、あるループ状にぐるぐる回るプログラムがあったとします。このプログラムが１周するのに10秒かかるとしますと、このプログラム内のある処理、たとえばキー入力の処理は10秒に１回しか行われないことになります。もし仮にこのゲームがリアルタイムのアクションゲームだとすると、キーの読み込みに10秒もかかっていては、とても遊ぶことはできませんね。こんなときは、今プログラムがどの行を実行していても、キーが押されたらキーの読み込み処理に実行を移し、その処理が終了したらもとの行に戻るようなことができれば、たいへんありがたいものです。

実は割り込みというのは、このように、今プログラム中のどこを実行していても、ある要因（この場合はキーを押すこと）によってすぐさま、指定された処理に移行するような処理方式を言うのです。

MSX-BASICでは、この割り込み処理が他のBASICと比較して非常に強力なものとなっており、キーによる割り込みの他にスプライトの衝突による割り込み、時間による割り込みなどがサポートされています。**表４**にMSX-BASICの割り込み機能をまとめておきますので参考にしてください。

表４

ＭＳＸ－ＢＡＳＩＣの割り込み機能
ＭＳＸ－ＢＡＳＩＣには強力な割り込み機能がサポートされていますが、その書式はどれも同じになっています。 以下に、キーワードを××××として、割り込みに関する命令を一括して説明します
ＯＮ　××××　ＧＯＳＵＢ　行番号１、行番号２、行番号３、．．．．
割り込みによる飛び先を指定します 飛び先の行番号がいくつかあるのは、たとえば、ファンクション・キーによる割り込みの場合など、ファンクション・キーが幾つもありますので、どのキーが押された場合どの飛び先に飛ぶのかを決めるには、キーの数だけの飛び先が必要となるからです
××××　ＯＮ
割り込みを有効にします。つまり、指定された割り込みの処理を可能にします
××××　ＯＦＦ
割り込みを無効にします。つまり、割り込みを発生するような動作が行われても、割り込み処理は行いません
××××　ＳＴＯＰ

割り込みを一時、保留にします。再開は、××××　ONによって行われます

以下に、各キーワードの説明と使用方法を示します

１．ファンクション・キーによる割り込み
　キーワード＝ＫＥＹ

　Ｆ１、Ｆ２などのファンクション・キーを押すことによって発生する割り込みです
　ＯＮ　ＫＥＹ　ＧＯＳＵＢで指定する行番号は、それぞれ、Ｆ１、Ｆ２などの各ファンクション・キーに対応しています
　また、ＯＮ，ＯＦＦ，ＳＴＯＰでは、各キーごとに割り込みの状態を指定することが可能です。たとえば

ＫＥＹ（１）　ＯＮ
ＫＥＹ（２）　ＯＦＦ
ＫＥＹ（４）　ＳＴＯＰ

　などです

２．ジョイスティックの発射ボタン（トリガーボタン）による割り込み
　キーワード＝ＳＴＲＩＧ

　ジョイスティックのトリガーボタンまたは、本体のスペースキーを押すことに発生する割り込みです
　ＯＮ　ＳＴＲＩＧ　ＧＯＳＵＢで指定する行番号は、それぞれのジョイスティックまたは、本体のスペースキーに対応しています
　また、ファンクション・キーによる割り込みと同じく、ＯＮ，ＯＦＦ，ＳＴＯＰでは、格ジョイスティックごとに指定が可能です。たとえば

ＳＴＲＩＧ（０）　ＯＮ

ＳＴＩＲＧ（１）　ＯＦＦ

　などです

３．ストップ・キーによる割り込み
　キーワード＝ＳＴＯＰ

ストップ・キーが押されることによって発生する割り込みです

４．スプライトの衝突による割り込み
　キーワード＝ＳＰＲＩＴＥ

　スプライトの衝突によって発生する割り込みです。ゲームによく使用します

５．時間による割り込み
　キーワード＝ＩＮＴＥＲＶＡＬ

　ある決められた時間になると発生する割り込みです
　ＯＮ　ＩＮＴＥＲＶＡＬ　ｚｚｚｚ　ＧＯＳＵＢで飛び先を指定しひますが、ｚｚｚｚの値は６０分の１秒Ｘｚｚｚｚとなります。たとえば

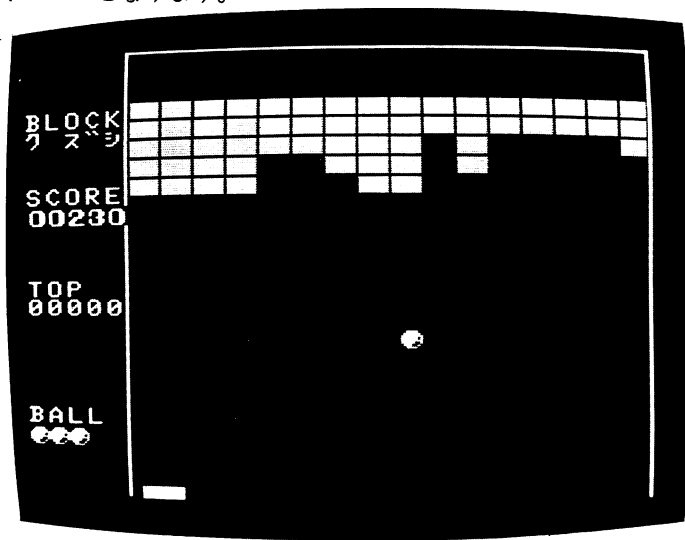
ＯＮ　ＩＮＴＥＲＶＡＬ　１２０　ＧＯＳＵＢ

の場合は、（１／６０）Ｘ１２０＝２秒ごとに割り込みが発生することを示します

とかくBASICで作ったアクションゲームは遅くてとても遊べない、などと言う人がいますが、この割り込み機能を十分使いこなせば、そうとうおもしろいプログラムを作ることができますよ。

さあ！始めよう！

ブロックくずし、と聞いて知らない人はいないと言えるほどポピュラーなゲームです。ようするにブロックと自分（パドル）の間をひとつのボールが飛びまわっていて、ブロックにボールがあたるとそのブロックが消えて得点になり、逆に自分の方にボールが来たときはパドルを動かして打ち返さなければならないわけです。

もし打ち返せなかったらボールが１つずつ減っていき、ゼロになったらゲームオーバーとなります。

遊び方

とにかくまず遊びたい人は、**リスト1**を注意深く入力して、カセットにセーブしておいてください。というのも、このプログラムは一部機械語を使用しており、1480行以後の機械語データの入力をミスしますと、あわれマイコンは暴走し、入力したプログラムはすべてパーになってしまいます。ちゃんとリストを見くらべて、正確に入力できたらRUNします。

STRING(J)

まずタイトルが出て、ジョイスティックで遊ぶか、MSX本体のカーソルキーで遊ぶか聞いてきますので、ジョイスティックの場合はY、カーソルキーの場合はNを入力してください。なお、ジョイスティックは、No.1側に接続してください。

これに答えると次にトリガーボタンを押すように指示が出ますので、本体キーの場合はスペースキー、ジョイスティックの場合はジョイスティックに付いているボタンを押すと、ボールが発射されゲームスタートとなります。

あとは必死になってボールを打ち返すのみです。例によって、ボールが天井に当たるとパドルの長さが半分となり、ますます難しくなります。さらにボールがパドルのどの位置に当たったかによって、ボールの反射角およびスピードが変化しますので、かなり本物に近くなっています。BASICでやるには、この程度が限度でしょう。

得点はブロック1個につき10点で、何点取ってもボールの数は増えません。あしからず。ゲームオーバーになりますと、再ゲームするかどうか聞いてきますので、再ゲームする場合はトリガーボタンを押してください。このときにハイスコアの更新も行われます。

プログラム

次にプログラムについて解析しましょう。まずこのプログラムを作成するとき一番問題になったのが、高解像度グラフィック（**SCREEN 2**）でのテキスト表示の遅さです。

SCREEN 2

このゲームでは、ブロックにボールが当たるごとにスコアの更新を行っているのですが、BASICでこれを行うとあまりの遅さに、ボールが一瞬止ってしまうという現象が発生しました。

あれやこれやいろいろとやってみたのですが、どうもBASICではうまくゆかず、ついに機械語で表示することにしました。リストを見てもらえばわかるように、1480行から1700行までのデータが機械語プログラムで、プログラムの起動時に260～300行でメモリー内に機械語プログラムを作っています。

USR 0
USR 1
USR 2
DEFUSR
CLEAR

機械語プログラムはD000番地からD0AF番地までですが、内部で3つに分かれており、それぞれD000番地（**USR 0**）が点数を0にするプログラム、D010番地（**USR 1**）が点数に10点たして表示するプログラム、そしてD021番地（**USR 2**）が現在の点数を表示するプログラムです。これら機械語プログラムの宣言（**DEFUSR**）および機械語プログラム用のメモリーの確保（**CLEAR**）は、100行、110行で行っています。このプログラムによって、ようやく遊ぶことのできるレベルになりました。

次にBASICの部分についてお話しましょう。

ボールおよびパドルは、スプライトを使用しています。このスプライトは、ゲームを作成するのに強力な命令で、任意の図形をスムーズにかつ高速に移動させることが可能です。スプライトの初期化は350～480行で行っており、**SPRITE$ (0)** にボールの図形データ **SPRITE$ (1)** にパドルの図形データを入れています。

SPRITE$ (0)
SPRITE$ (1)
LINE

ブロックは**LINE**命令で枠を書き、ぬりつぶすことによって作成しています（620～690行)。ボールやパドルの移動は単にX軸方向、Y軸方向に移動量をたして行き、スプライトを表示すれば良く、ボールのスピードを変化させるには，この移動量を変化すれば良いわけです。このボールの移動量はXS, YSに、ボールの位置はXB，YBに、パドルは横方向にしか動けないのでY軸方向は固定で、X軸は

XPにそれぞれ入っています。

次が一番のノウハウである衝突の処理です。ボールは壁、パドル、ブロックの3つのものに衝突するわけですが，この処理が遅いとボールの動きがにぶくなり、ゲームがおもしろくなくなってしまいます。そこでこのプログラムでは、この3つのものと衝突した場合の判断をそれぞれ異なる方法で行い、高速化を実現しています。

まず壁との衝突ですが、これは壁の存在する座標がX，Y軸共に分けて処理しやすいので、座標によって判断することにしました。950～990行あたりがこの処理を行っている部分です。次にパドルとの衝突ですが、スプライト同士の衝突についてはMSXでは**ON SPRITE GOTO (GOSUB)**というとても便利な命令があるのでこれを使用しました。この命令は、いちいち2つのスプライトの座標を計算して衝突したかの判断をしなくても、2つ以上のスプライトが衝突すると自動的に指定した飛び先またはサブルーチンに分岐してくれるものです。330行、800行でこの制御を行っています。

ON～SPRITE～GOTO (GOSUB)

最後にブロックとの衝突ですが、これは**POINT**命令でそこがブロックの色かどうかを見て、ブロックとの衝突を判断しています。**POINT**命令は、任意の座標の点の色を調べるための命令で、ボールの移動先は何もないか、ブロックかの2つの状態しかありませんので、これによって簡単に判断ができるわけです。

POINT

なお、ブロックの消去は**PAINT**命令で黒くぬりつぶすことによって行っています。

PAINT

このプログラムは16Kバイトシステム以上で動作し、ディスクシステムでも使用できます。

フロッピーディスクをお使いの方へ

あなたがフロッピーディスクを使っているならば，プログラムの以下のリストのように一部訂正してください。これで，バッチリ。

```
 100 MAXFILES=3:CLEAR 900,&HCFFF:DEFINT A-Z:DEFUSR=&HD000
 110 DEFUSR1=&HD010:DEFUSR2=&HD021:OPEN"GRP:"FOR OUTPUT AS #1
 260 AD=&HD000:RESTORE 1490
1490 DATA 21,51,D0,3E,06,36,00,23
1510 DATA 21,52,D0,06,05,7E,3C,27
1530 DATA F4,21,51,D0,06,06,7E,87
1540 DATA 87,87,C5,E5,21,57,D0,16
```

リスト1(その1) ブロックくずし

```
10 ' Block くずし
20 ' by Tamae Tama
30 ' debug and advice by Haruka takagi
40 ' copyright 1984 (C) NATS
50 ' copyright 1984 (C)
60 '   Seibundo-Shinkousha publishing
70 '
80 ' HENSU & SYSTEM INIT
90 '
100 MAXFILES=3:CLEAR 900,&HDFFF:DEFINT A-Z:DEFUSR=&HE00
    0
110 DEFUSR1=&HE010:DEFUSR2=&HE021:OPEN"GRP:"FOR OUTPUT
    AS #1
120 XS=0:YS=0:'BALL SPEED X,Y
130 XB=0:YB=0:'BALL X,Y
140 XP=0:'     PADL X
150 TP=0:'     HI- SCORE
160 SCREEN 1:CLS
170 PRINT "******************************";
180 PRINT "*          Block くずし          *";
190 PRINT "*                            *";
200 PRINT "*    by T.Tama & H.takagi    *";
210 PRINT "*    copyright 1984 (C)      *";
220 PRINT "*    Seibundo-Shinkousha     *";
230 PRINT "******************************";
240 PRINT:PRINT
250 INPUT "     JOYSTICK ?(Y/N)";I$
260 AD=&HE000:RESTORE 1490
270 READ D$:IF D$="end" THEN GOTO 310
280 D=VAL("&H"+D$):POKE AD,D
290 AD=AD+1
300 GOTO 270
310 J=0:IF I$="Y" OR I$="y" THEN J=1
320 SCREEN 2,0,0:KEY OFF
330 ON SPRITE GOSUB 1280
340 BALL=4:SCENE=-1:HIT=0:U=USR(0)
```

リスト1（その2） ブロックくずし

```
350 '
360 '  SPRITE INIT
370 '
380 RESTORE
390 SCREEN 2,0
400 S0$="":S1$=""
410 FOR I=0 TO 7
420 READ R
430 S0$=S0$+CHR$(R)
440 S1$=S1$+CHR$(255):NEXT
450 SPRITE$(0)=S0$
460 SPRITE$(1)=S1$
470 SCENE=SCENE+1
480 CL=(SCENE AND 3)*2+8
490 '
500 '  DRAW GAME STAGE
510 '
520 COLOR15,1,0:CLS:PRESET(8,24):PRINT#1,"BLOCK"
530 PRESET(8,32):PRINT#1,"ｸ ｽﾞｼ"
540 PRESET(8,56):PRINT#1,"SCORE"
550 PRESET(8,65):PRINT#1,"¥¥¥¥¥"
560 PRESET(8,96):PRINT#1,"TOP"
570 TP$=RIGHT$("0000"+RIGHT$(STR$(TP),LEN(STR$(TP))-1),
    5)
580 PRESET(0,104):PRINT#1,USING" &   &";TP$
590 U=USR2(0)
600 PRESET(8,152):PRINT#1,"BALL"
610 LINE(47,0)-(240,192),15,B
620 LINE -(47,192),0
630 LINE(48,20)-(238,60),CL,BF
640 FOR I=20 TO 60 STEP 8
650 LINE(48,I)-(239,I),1
660 NEXT
670 FOR I=48 TO 239 STEP 12
680 LINE(I,20)-(I,60),1
690 NEXT
700 XP=100:HIT=0
710 GOSUB1210
720 XS=1:YS=-4:XB=100:YB=100
730 PUT SPRITE 1,(XP,185),12,1:PUT SPRITE2,(XP+8,185),1
    2,1
```

リスト1(その3) ブロックくずし

```
 740 '
 750 '  START ?
 760 '
 770 PSET (70,100),0:PRINT #1,"ﾄﾘｶﾞ  ﾎﾞﾀﾝｦ ｵｼﾃｸﾀﾞｻｲ":F=0
 780 IF STRIG(J)=0 THEN 780
 790 LINE(70,100)-(230,108),0,BF
 800 BEEP:SPRITE ON
 810 GOSUB 1210
 820 '
 830 '   PADL MOVE
 840 '
 850 FOR HIT=HIT TO 79
 860 IF 3=STICK(J) AND 223+8*F>XP THEN XP=4+XP ELSE 890
 870 PUT SPRITE2,(8+XP,185+24*F),12,1
 880 PUT SPRITE1,(XP,185),12,1:GOTO 950
 890 IF 7=STICK(J) AND 48<XP THEN XP=XP-4 ELSE 950
 900 PUT SPRITE1,(XP,185),12,1
 910 PUT SPRITE2,(8+XP,185+24*F),12,1
 920 '
 930 '   BALL MOVE
 940 '
 950 XB=XB+XS:YB=YB+YS
 960 IF 231<XB THEN XS=-ABS(XS):PLAY"L64A":GOTO 980
 970 IF 48>XB THEN XS=ABS(XS):PLAY"L64A"
 980 IF 1>YB THEN YS=ABS(YS):PLAY"L64A":Y=5:F=1:PUTSPRIT
     E 2,(0,209),0,0
 990 IF 200<YB THEN1100
1000 PUT SPRITE0,(XB,YB),12,0
1010 '
1020 '   HIT ?
1030 '
1040 IF CL-POINT(3+XB,3+YB) THEN 860
1050 PLAY"L64B":PAINT(3+XB,3+YB),1
1060 YS=-YS:U=USR1(0):NEXT:GOTO 380
```

リスト1(その4) ブロックくずし

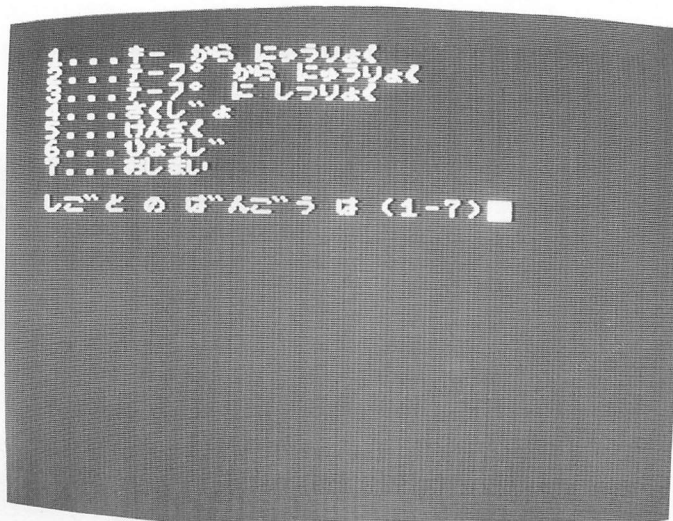
```
1070 '
1080 ' BALL OUT
1090 '
1100 PLAY"L5O1A":BALL=BALL-1
1110 FOR I=0 TO 1000:NEXT
1120 IF BALL>0 THEN 720
1130 PRESET(80,80):PRINT #1,"GAME OVER":SC=(HIT+SCENE*8
     0)*10
1140 IF TP<SC THEN TP=SC
1150 PRESET(80,96):PRINT #1,"REPLAY : ﾄﾘｶﾞ ﾎﾞﾀﾝ"
1160 IF STRIG(J)=0 THEN 1160
1170 GOTO 340
1180 '
1190 ' NOKORI BALL WRITE
1200 '
1210 C=12:FOR I=1 TO 5
1220 IF I=BALL THEN C=1
1230 PUT SPRITE I+2,(8*I,160),C,0
1240 NEXT:RETURN
1250 '
1260 ' SPRITE WARIKOMI
1270 '
1280 SPRITE OFF:BEEP
1290 ON 3+F+(XB-XP)/4 GOTO 1300,1310,1340,1340,1330,132
     0
1300 XS=-5:GOTO 1340
1310 XS=-2:GOTO 1340
1320 XS=5:GOTO 1340
1330 XS=2
1340 YS=-ABS(YS):XB=XB+XS:YB=YB+YS
1350 PUT SPRITE 0,(XB,YB),12,0
1360 SPRITE ON:RETURN
```

リスト1(その5) ブロックくずし

```
1370 '
1380 ' BALL SPRITE DATA
1390 '
1400 DATA &B00111100
1410 DATA &B01111110
1420 DATA &B11111111
1430 DATA &B11111101
1440 DATA &B11111111
1450 DATA &B11111001
1460 DATA &B01110010
1470 DATA &B00111100
1480 ' machine language data
1490 DATA 21,51,E0,3E,06,36,00,23
1500 DATA 3D,20,FA,C9,00,FE,41,20
1510 DATA 21,52,E0,06,05,7E,3C,27
1520 DATA E6,0F,77,20,04,23,05,20
1530 DATA F4,21,51,E0,06,06,7E,87
1540 DATA 87,87,C5,E5,21,57,E0,16
1550 DATA 00,5F,19,E5,21,F8,07,CB
1560 DATA 20,CB,20,CB,20,48,06,00
1570 DATA 09,54,5D,E1,01,08,00,CD
1580 DATA 5C,00,E1,C1,23,05,20,D6
1590 DATA C9,00,00,00,00,00,00,00
1600 DATA 3C,46,46,46,46,46,3C,00
1610 DATA 1C,3C,7C,1C,1C,1C,7E,00
1620 DATA 3C,46,46,1C,30,60,7E,00
1630 DATA 3C,46,46,0C,46,46,3C,00
1640 DATA 06,0E,16,26,7E,06,06,00
1650 DATA 7C,60,60,7C,06,46,3C,00
1660 DATA 1C,20,40,7C,46,46,3C,00
1670 DATA 7E,46,46,0C,18,18,18,00
1680 DATA 3C,46,46,3C,46,46,3C,00
1690 DATA 3C,46,46,3E,06,06,3C,06
1700 DATA 00,FF,00,FF,00,FF,end
```

ホームデータベース

みなさん、自分のもっているプログラム（カセットテープに入っているもの）やレコード、友だちの住所などを整理するとき、どのように行っていますか？　もちろん、ノートに付けて行く方法もありますが、それでは現在の情報化時代に乗り遅れてしまいますし、せっかくマイコンがあるんですから、これにやらせてみようではありませんか。

このプログラムは、言うならばコンピューターによるメモ帳で、あらかじめ入れておいたデータ（たとえば、レコードの名前、値段、買った日、保存場所）に対して、キーワード（捜すのにカギとなる文字）を入力することにより、目的の情報をすばやく捜し出すことが可能なものです。

使い方

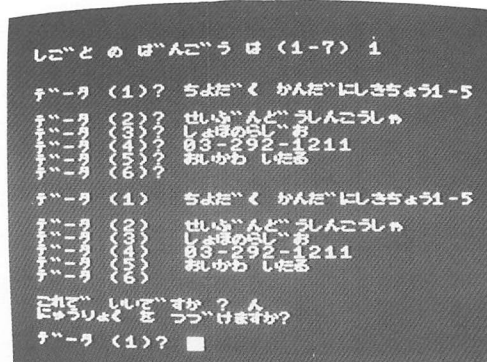
まず**リスト2**を正確に入力してください。このプログラムはずいぶんとつまっており、ごちゃごちゃして見にくいのですが、ゆったりと見やすくプログラムを作ると、プログラムの大きさが大きくなってしまい、かんじんのデータ領域が小さくなってしまうので泣く泣く、こうなってしまったわけです。

RUN

リスト2を入力したら、**RUN**とすると一定時間タイトルが表示され、メニューが表示されます。次にそれぞれのモードについて解説しましょう。

1．キーボードからのデータ入力

1を押すとキーボードからのデータ入力モードとなります。各データは6つの項目を持っており、それぞれ何を入れるかは自分で決めることができます。また、内容としては文字、数字が使用できますが、特殊文字として「,（カンマ）」および「"（ダブルコーテーション）」は使用できませんので注意してください。

データの数は100個まで使用できるように配列を宣言していますが、RAMの容量や1つデータの文字数によって、100件では多すぎてエラーとなる場合や、100件入れてもメモリー上に余裕ができてしまう場合がありますので、これは自分で調整してみてください。なお、データは追加されていく形で入りますので、前のデータが消されてしまう、といった心配はありません。

2．テープからのデータ入力

まず、キーボードから入力して作ったデータを3のテープへの出力を使って、カセットテープに出力しておきます。そして、この次以後は、いちいちデータをもう一度入力しなくても、この命令によりカセットテープからデータを入力することができるわけです。

また、2つの別々に作っておいたデータを、この命令で読み込むことによって、1つにすることが可能です（アペンド機能)。

3．テープへのデータ出力

現在メモリー上にあるデータをカセットテープにすべて出力します。逆にテープからデータを読み込むのは2の命令を使用します。

4．データの削除

データの入力のときにミスをして入れてしまったデータや不要になったデータを消すときに使用します。

データの消しかたは、キーワードを入力することにより、次々と該当するデータを表示しますので、消したいデータが見つかったらｙを押しますと、データを消すことができます。

5．データの検索

PRINT
LPRINT

まず、検索するデータの各項目のキーワードを聞いてきますので入力して行きます。どうでもよい項目については、リターンキーのみ入力すれば次の項目に進んでくれます。キーワードの入力を終了すると、この条件に基き、データの中から該当するものを次々と捜し出して表示を行います。なお、このプログラムでは画面のみでプリンターなどには出力できませんが、840行の**PRINT**文を**LPRINT**文に変更することによってプリンターにも出力することが可能です。

6．全データの表示

現在メモリー内に存在する全データの表示を行います。つまり、データの検索で、キーワードをすべてリターンキーのみしか押さなかったときと同じ働きをするわけです。

7．終了

ON ERROR
GOTO 0

このプログラムの実行を終了します。ここでは、エラートラップの解除（**ON ERROR GOTO 0**）を行っていますので、必ずこの命令によってプログラムを終了させ、ストップキーでは止めないようにしてください。

プログラム

プログラムの処理の中心は、配列Kに入っているデータの書き換え（入力）、交換（削除）および表示が主なもので、現在のデータの個数は変数Pに入っています。

SWAP

特に複雑な処理を行っている部分はありませんが、データの削除方法として **SWAP** 命令を使用した変数の入れ替え方式を使用している点が、少し高度な処理に入るほうでしょうか。

ON J GOTO

各処理は、330行の **ON J GOTO** 文によって飛んでおり、

350行＝キーからのデータ入力
510行＝テープからのデータ入力
570行＝テープへのデータ出力
630行＝データの削除
750行＝データの検索
830行＝全データの表示
880行＝終了

となっています。

OPEN "CAS : DATA"
OPEN "DATA"

このプログラムは16Kバイト以上のシステムで使用可能で、ディスクシステムでも使用可能です。なお、ディスクシステムでは、**OPEN "CAS : DATA"** の部分を **OPEN "DATA"** にするとテープではなく、ディクスに対してデータの読み書きを行いますので、テープよりずっと速く処理が行えます。

```
 10 ' Home data-base program
 20 ' original by Haruka Takagi
 30 ' by Mika Tada
 40 ' by Haruka Takagi
 50 ' copyright 1984 NATS
 60 ' copyright 1984 (C)
 70 '     Seibundo-Shinkousha publishing
 80 DEFINT A-Z:P=0
 90 SCREEN 1:CLS:COLOR 15,0,0:KEY OFF
100 ON ERROR GOTO 930
110 PRINT "*****************************";
120 PRINT "*   Home Data-base program  *";
130 PRINT "*                           *";
140 PRINT "*    by M.Tada & H.takagi   *";
150 PRINT "*    copyright 1984 NATS    *";
160 PRINT "*    copyright 1984 (C)     *";
170 PRINT "*    Seibundo-Shinkousha    *";
180 PRINT "*****************************"
190 FOR TM=0 TO 5000:NEXT
200 DIM D$(5,100)
210 CLS:PRINT "1...キー から にゅうりょく"
220 PRINT "2...テーフ゜ から にゅうりょく"
230 PRINT "3...テーフ゜ に しつりょく"
240 PRINT "4...さくし゛ょ"
250 PRINT "5...けんさく"
260 PRINT "6...ひょうし゛"
270 PRINT "7...おしまい"
280 PRINT :PRINT "しこ゛と の は゛んこ゛う は (1-7)";
290 J$=INPUT$(1):J=VAL(J$)
300 IF J>7 OR J<1 THEN 290
310 PRINT J
320 PRINT
330 ON J GOTO 350,510,570,630,750,830,880
```

リスト2(その2) ホームデータベース

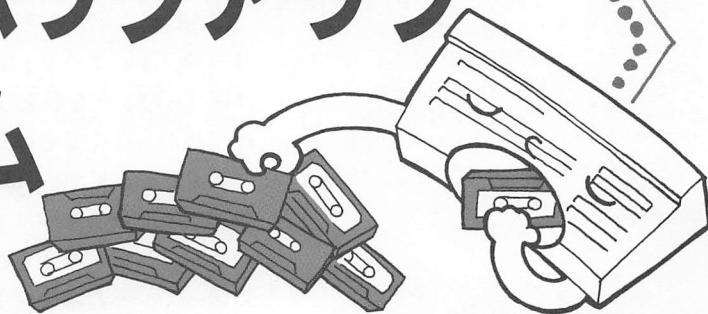
```
340 'data input
350 P=P+1:PRINT
360 FOR I=0 TO 5
370 PRINT USING "デ゛ータ (#)";I+1;:INPUT D$(I,P):NEXT:PRIN
    T
380 FOR I=0 TO 5:PRINT USING "デ゛ータ (#)  ";I+1;:PRINT D$
    (I,P):NEXT:PRINT
390 PRINT "これて゛ いいて゛すか ";
400 INPUT A$:IF A$="N" OR A$="n" THEN 410 ELSE 480
410 PRINT "なんは゛ん の テ゛ータ を なおしますか ( 1-6 ) ";
420 INPUT N
430 IF N>6 OR N<0 THEN 420
440 PRINT USING "デ゛ータ (#)  ";N;:PRINT D$(N-1,P);"   これて
    "すか";
450 INPUT A$
460 IF A$="Y" OR A$="y" THEN 470 ELSE IF A$="N" OR A$="
    n" THEN 410 ELSE 450
470 PRINT USING "デ゛ータ (#)";N;:INPUT D$(N-1,P):PRINT :GO
    TO 380
480 INPUT "にゅうりょく を つつ゛けますか";A$:IF A$="n" OR A$="N" THE
    N GOTO 210 ELSE GOTO 350
```

リスト2(その3) ホームデータベース

```
490 'read from tape
500 '
510 PRINT :INPUT "カセットテープのじゅんびはいいですか ";A$:IF A$="
    N" OR A$="n" THEN 210
520 OPEN "CAS:DATA" FOR INPUT AS#1
530 INPUT #1,PA:IF PA+P>100 THEN CLOSE #1:PRINT :PRINT
    "データが おおすぎます !!":GOTO 210
540 FOR L1=1 TO PA:FOR L2=0 TO 5:INPUT #1,D$(L2,P+L1):N
    EXT :NEXT :CLOSE #1:P=P+PA
550 PRINT "データ にゅうりょく かんりょう":PRINT:GOTO 850
560 'write to tape
570 IF P=0 THEN PRINT :PRINT "でーたが ありません !! ":PRINT :
    GOTO 850
580 PRINT :INPUT "カセットテープのじゅんびはいいですか ";A$:IF A$="
    N" OR A$="n" THEN 210
590 OPEN "CAS:DATA" FOR OUTPUT AS#1:PRINT #1,P
600 FOR L1=1 TO P:FOR L2=0 TO 5:PRINT #1,D$(L2,L1):NEXT
     :NEXT :CLOSE #1
610 PRINT :PRINT "データを テープに しつりょく しました !!":PRINT :BEE
    P:GOTO 850
620 'delete data
630 IF P=0 THEN 830
640 KS=1
650 GOSUB 890
660 GOSUB 900
670 IF KL=1 THEN 700
680 KS=KS+1:IF KS=<P THEN 660
690 BEEP:PRINT :PRINT "データが なくなりました !!":PRINT :GOTO 8
    50
700 PRINT :INPUT "この データを けしますか ";A$:IF A$="Y" OR A$="
    y" THEN 710 ELSE 680
710 FOR L1=KS TO P-1:FOR L2=0 TO 5:SWAP D$(L2,L1+1),D$(
    L2,L1):NEXT :NEXT
720 P=P-1
730 PRINT :PRINT "データを けしました !! ":PRINT :GOTO 850
```

リスト2(その3) ホームデータベース

```
740 'search data
750 IF P=0 THEN 830
760 KS=1
770 GOSUB 890
780 GOSUB 900
790 KS=KS+1:IF KS>P THEN 690
800 IF KL=0 THEN 780
810 GOTO 780
820 'display all data
830 IF P=0 THEN PRINT :PRINT :PRINT "また゛ テ゛ータ か゛ ありません
    !! ":BEEP:PRINT :GOTO 850
840 PRINT :FOR L1=1 TO P:FOR L2=0 TO 5:PRINT USING "テ゛ー
    タ (#)";L2+1;:PRINT D$(L2,L1):NEXT :FOR W=0 TO 500:N
    EXT W:PRINT :NEXT
850 PRINT "つき゛の しょりを するために なにか キー を おして くた゛さい !"
860 IF INKEY$="" THEN 860 ELSE 210
870 'end of job
880 PRINT :PRINT :PRINT "こ゛くろうさまて゛した":ON ERROR GOTO 0:E
    ND
890 PRINT :FOR L=0 TO 5:PRINT USING "テ゛ータ (#) キーワート゛";L
    +1;:KW$(L)="":INPUT KW$(L):NEXT:RETURN
900 KA=1:FOR L1=0 TO 5:KK=INSTR(D$(L1,KS),KW$(L1)):KA=K
    A*KK:NEXT
910 IF KA=0 THEN KL=0:RETURN
920 PRINT :PRINT :FOR L1=0 TO 5:PRINT USING "テ゛ータ (#)
     ";L1+1;:PRINT D$(L1,KS):NEXT :KL=1:RETURN
930 CLS
940 PRINT "エラー か゛ はっせい しました!!!"
950 PRINT "もういちと゛ やりなおしてくた゛さい"
960 INPUT "なにか キーを おしてくた゛さい"
970 RESUME 210
```

スーパーバックアッププログラム

みなさんの周りにも、MSX マシンを持っている人が多いことと思いますが、ちょっと人の持っているカセットテープのプログラムをもらいたいな、と思ったときも、一本にたくさんはいっているとなかなかめんどうですね。

こんなときには、いまはやりのダブルデッキなんかを使って移すこともできるのですが、パソコンのプログラムやデータの信号（音）というものは、実に微妙で、ちょっとドロップアウトがあったり、波形がなまっていたりすると、すぐにエラーとなってしまいます。

ほんとでしたら、プログラムひとつひとつを本体に読み込んでからセーブするのがベストなのですが、これは手間がかかり、時間もかかるのでやりたくないですね。でも、確実に移しておきたいなんて、ちょうしの良いことができないでしょうか。

このプログラムは、これを可能にしたもので、使い方は実に簡単。ダブルデッキで移すのと同じ要領でうまくバックアップを使うことができます。

```
Cassette copy program for MSX
Version ][
by Haruka Takagi
copyright 1984 NATS

PHASE invert (y/n)n
SOUND output (y/n)y
```

プログラムの原理

MSX がテープにプログラムやデータをセーブするときは、メモリーの「1」と「0」を音に変えてセーブするわけですが、この音は図 1a のような形をしています。ちょうど正弦波（わからない人は、わからなくてもけっこうです）が少しゆがんだような形ですね。

さてこの信号が、MSX のカセット読み込み回路に入ると、図 1b のように方形の波形に変えられます。つまり、「0」と「1」に波形が変形されたと考えられます。そこで、この波形をすごく短い時間で切って、その時の「1」か「0」かを逆にカセットの出力回路に送ってやれば、ほぼ似たような波形が得られることになります（図 1c参照）。

まず、2台のカセットテレコを用意し、片方を再生しその信号を MSX に入力。このプログラムによってきれいになった信号が再びカセット接続ケーブルにもどって、もう一台のテレコに出力して記録すれば、ダブルデッキよりも良好なバックアップを作ることができます。

ただし、バックアップをくり返すと、テレコのワウ・フラッターの影響が大きくなりエラーが発生しますので、この方法でのバックアップは1回が限度と考えるのが良いでしょう。それに個人で使用する場合、バックアップは1本あればまずだいじょうぶですからね。

2・4kHzの方形波を入力し(下)、出力(赤プラグ)よりカセットテープに記録する波形が上。

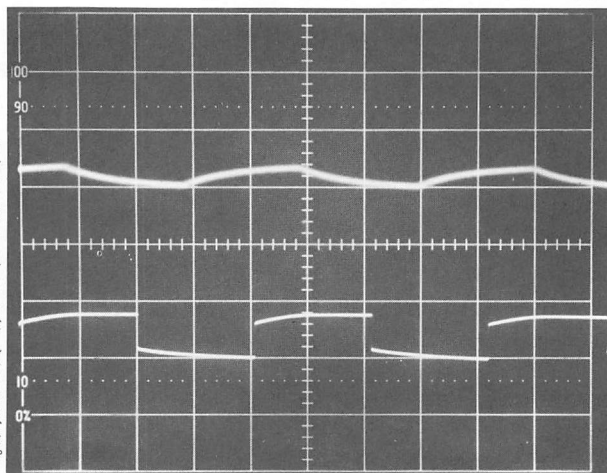
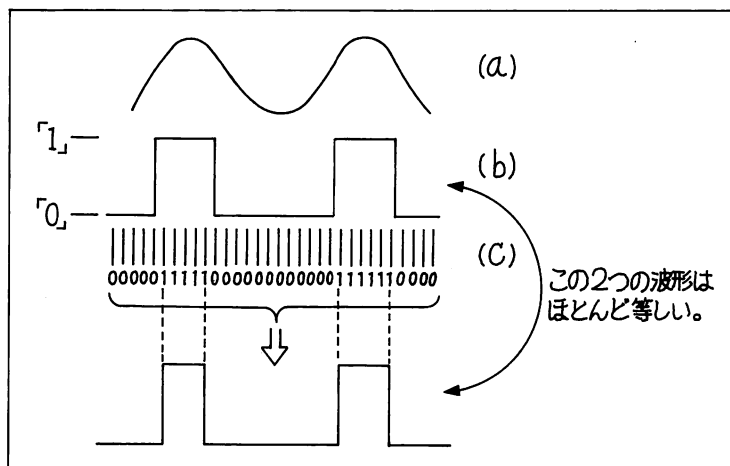



図 1

プログラム

DATA

リスト3aにスーパーバックアッププログラムのリストを示します。一見すべてBASICで書かれているように見えますが、実は、BASICでは単にメモリー上に機械語プログラムを作っているだけなのです。機械語のプログラムは220行以後の**DATA**文で構成されています。

なぜ機械語を使うかと言うと、先ほど話しましたが、カセットから入力される波形をすごく短い時間で切るには、BASICでは遅すぎるためなのです。また、機械語そのものは人間にわかりにくい(ただの数字です)ので、アセンブラと呼ばれる言語を使用し、これを機械語に変換するわけです。このプログラムをアセンブラで書いたものを**リスト3b**に示します。

ちんぷんかんぷんの人も多いと思いますが、MSXをほんとうに使いこなすためには、必ずアセンブラが必要となります。みなさんがアセンブラを勉強し始めたら、必ずこのリストが役に立ちますので、それまでこの本はだいじに保存してくださいね。

プログラムの使い方

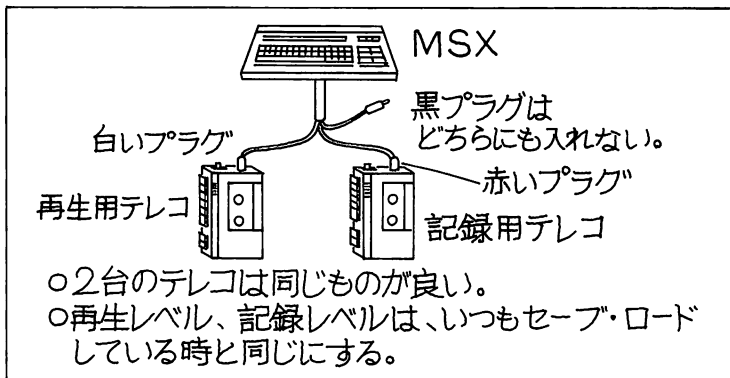
まず、**リスト3a**を入力し、必ずセーブしておきます。このプログラムは機械語を使用しているため、**DATA** 文のデータを1つでも間違うと暴走するからです。

次に、**図2**のようにMSXと2台のテレコを継ぎます。2台のテレコは、できれば同じものがベストです。そして、再生用テレコにマスターテープを、記録用テレコに新しいテープを入れ、このプログラムをRUNします。

タイトルが出てしばらくすると、PHASE invert (y/n)と聞いてきますので、2台のテレコの位相関係が逆の場合はy、同じ場合はnを入れてください。わからない場合は、バックアップしたテープがエラーを起したりする場合、yかnを適当に入れかえてちゃんと読めるほうにしてください。

次にSOUND output (y/n) と聞いてきますので、カセットの音をMSXのオーディオ出力端子から出したい場合はy、出したくない場合はnを入力してください。

これで準備が完了しました。あとは再生用テレコを再生し、記録用テレコを記録(録音)にすれば自動的にカセット1本分をバックアップしてくれます。なお、このプログラムは一度動き出すと電源を切る以外に止めることはできません。また、このプログラムは16Kバイト以上のシステムで使用可能で、ディスクがつながっていてもOKです。

図2

```
 10 ' Super cassette back up program
 20 ' by Haruka Takagi
 30 ' copyright 1984 (C) NATS
 40 ' copyright 1984 (C)
 50 '  Seibundo-Shinkousha publishing
 60 MAXFILES=0:CLEAR 10,&HF1FF
 70 SCREEN 0:CLS
 80 PRINT "****************************************";
 90 PRINT "*     Super cassette back up program   *";
100 PRINT "*     by Haruka Takagi                 *";
110 PRINT "*     copyright 1984 (C) NATS          *";
120 PRINT "*     copyright 1984 (C)               *";
130 PRINT "*    Seibundo-Shinkousha publishing    *";
140 PRINT "****************************************";
150 PRINT:PRINT:PRINT
160 PRINT "Please wait for a moment!!!"
170 AD=&HF200
180 READ D$:IF D$="end" THEN GOTO 220
190 D=VAL("&H"+D$)
200 POKE AD,D:AD=AD+1
210 GOTO 180
220 DEF USR=&HF200
230 A=USR(0)
```

リスト3a(その2) スーパーバックアップ

```
240 'machine language data
250 '&hF200 -> &hF300
260 '
270 DATA AF,32,FE,F2,11,7F,F2,CD
280 DATA 76,F2,11,D5,F2,CD,76,F2
290 DATA CD,9F,00,F5,CD,A2,00,11
300 DATA FB,F2,CD,76,F2,F1,FE,6E
310 DATA 28,0E,FE,79,20,E4,3E,2F
320 DATA 32,5D,F2,3E,80,32,FE,F2
330 DATA 11,E8,F2,CD,76,F2,CD,9F
340 DATA 00,F5,CD,A2,00,11,FB,F2
350 DATA CD,76,F2,F1,FE,6E,20,07
360 DATA 3E,20,32,65,F2,18,04,FE
370 DATA 79,20,DD,21,FE,F2,F3,3E
380 DATA 0E,D3,A0,DB,A2,00,E6,80
390 DATA BE,28,F4,A7,3E,A0,28,07
400 DATA D3,AA,3E,80,77,18,E8,3E
410 DATA 00,D3,AA,77,18,E1,1A,A7
420 DATA C8,CD,A2,00,13,18,F7,0C
430 DATA 43,61,73,73,65,74,74,65
440 DATA 20,63,6F,70,79,20,70,72
450 DATA 6F,67,72,61,6D,20,66,6F
460 DATA 72,20,4D,53,58,0D,0A,56
470 DATA 65,72,73,69,6F,6E,20,5D
480 DATA 5B,0D,0A,62,79,20,48,61
490 DATA 72,75,6B,61,20,54,61,6B
500 DATA 61,67,69,0D,0A,63,6F,70
510 DATA 79,72,69,67,68,74,20,31
520 DATA 39,38,34,20,4E,41,54,53
530 DATA 0D,0A,0D,0A,00,50,48,41
540 DATA 53,45,20,69,6E,76,65,72
550 DATA 74,20,28,79,2F,6E,29,00
560 DATA 53,4F,55,4E,44,20,6F,75
570 DATA 74,70,75,74,20,28,79,2F
580 DATA 6E,29,00,0D,0A,00,00,FF
590 DATA end
```

リスト3b(その1)

```
 10:                   ;Cassette copy program for "MSX"
 20:                   ;Version ][
 30:                   ;by Haruka Takagi
 40:                   ;copyright 1984 NATS
 50:                   ;
 60:  009F =           CHGET     EQU     009FH
 70:  00A2 =           CHPUT     EQU     00A2H
 80:  00A0 =           PSADR     EQU     0A0H
 90:  00A2 =           PSDAT     EQU     0A2H
100:  00AA =           PPC       EQU     0AAH
110:                   ;
120:  F200                       ORG     0F200H
130:  F200 AF          INIT:     XOR     A
140:  F201 32FEF2                LD      (CSTAT),A
150:  F204 117FF2                LD      DE,EMSG
160:  F207 CD76F2                CALL    MSG
170:  F20A 11D5F2      LOP1:     LD      DE,PHAS
180:  F20D CD76F2                CALL    MSG
190:  F210 CD9F00                CALL    CHGET
200:  F213 F5                    PUSH    AF
210:  F214 CDA200                CALL    CHPUT
220:  F217 11FBF2                LD      DE,CRLF
230:  F21A CD76F2                CALL    MSG
240:  F21D F1                    POP     AF
250:  F21E FE6E                  CP      'n'
260:  F220 280E                  JR      Z,NEXT
270:  F222 FE79                  CP      'y'
280:  F224 20E4                  JR      NZ,LOP1
290:  F226 3E2F                  LD      A,2FH
300:  F228 325DF2                LD      (CCC),A
310:  F22B 3E80                  LD      A,80H
320:  F22D 32FEF2                LD      (CSTAT),A
330:  F230 11E8F2      NEXT:     LD      DE,SOMSG
340:  F233 CD76F2                CALL    MSG
350:  F236 CD9F00                CALL    CHGET
360:  F239 F5                    PUSH    AF
370:  F23A CDA200                CALL    CHPUT
380:  F23D 11FBF2                LD      DE,CRLF
390:  F240 CD76F2                CALL    MSG
400:  F243 F1                    POP     AF
410:  F244 FE6E                  CP      'n'
420:  F246 2007                  JR      NZ,NNN
430:  F248 3E20                  LD      A,20H
440:  F24A 3265F2                LD      (OTD+1),A
450:  F24D 1804                  JR      MAIN
460:  F24F FE79      NNN:        CP      'y'
470:  F251 20DD                  JR      NZ,NEXT
480:                   ;
490:  F253 21FEF2      MAIN:     LD      HL,CSTAT
500:  F256 F3                    DI
```

リスト3b(その2)

```
510:  F257 3E0E        START:    LD      A,14
520:  F259 D3A0                  OUT     (PSADR),A
530:  F25B DBA2                  IN      A,(PSDAT)
540:  F25D 00          CCC:      NOP
550:  F25E E680                  AND     80H
560:  F260 BE                    CP      (HL)
570:  F261 28F4                  JR      Z,START
580:                   ;
590:  F263 A7                    AND     A
600:  F264 3EA0        OTD:      LD      A,0A0H
610:  F266 2807                  JR      Z,STATO
620:                   ;
630:  F268 D3AA                  OUT     (PPC),A
640:  F26A 3E80                  LD      A,80H
650:  F26C 77                    LD      (HL),A
660:  F26D 18E8                  JR      START
670:                   ;
680:  F26F 3E00        STATO:    LD      A,00H
690:  F271 D3AA                  OUT     (PPC),A
700:  F273 77                    LD      (HL),A
710:  F274 18E1                  JR      START
720:                   ;
730:  F276 1A          MSG:      LD      A,(DE)
740:  F277 A7                    AND     A
750:  F278 C8                    RET     Z
760:  F279 CDA200                CALL    CHPUT
770:  F27C 13                    INC     DE
780:  F27D 18F7                  JR      MSG
790:                   ;
800:  F27F 0C          EMSG:     DEFB    0CH
810:  F280 43617373              DEFM    'Cassette copy program '
820:  F296 666F7220              DEFM    'for MSX'
830:  F29D 0D0A                  DEFB    0DH,0AH
840:  F29F 56657273              DEFM    'Version ]['
850:  F2A9 0D0A                  DEFB    0DH,0AH
860:  F2AB 62792048              DEFM    'by Haruka Takagi'
870:  F2BB 0D0A                  DEFB    0DH,0AH
880:  F2BD 636F7079              DEFM    'copyright 1984 NATS'
890:  F2D0 0D0A0D0A              DEFB    0DH,0AH,0DH,0AH,00H
900:  F2D5 50484153    PHAS:     DEFM    'PHASE invert (y/n)'
910:  F2E7 00                    DEFB    00H
920:  F2E8 534F554E    SOMSG:    DEFM    'SOUND output (y/n)'
930:  F2FA 00                    DEFB    00H
940:  F2FB 0D0A00      CRLF:     DEFB    0DH,0AH,00H
950:  F2FE             CSTAT:    DEFS    1
960:  F2FF                       END

 F25D  CCC      009F  CHGET    00A2  CHPUT    F2FB  CRLF
 F2FE  CSTAT    F27F  EMSG     F200  INIT     F20A  LOP1
 F253  MAIN     F276  MSG      F230  NEXT     F24F  NNN
 F264  OTD      F2D5  PHAS     00AA  PPC      00A0  PSADR
 00A2  PSDAT    F2E8  SOMSG    F257  START    F26F  STATO
```

コンピュレコーダー バージョン2

CD（コンパクトディスク）やPCMなどデジタルオーディオを聞いたことのある人も多いことと思いますが、MSXを使って、このデジタルオーディオに挑戦してみたのが、このコンピュレコーダーです。

いつもは、プログラムやデータを読むのに使用しているカセット入力（白いプラグ）から、「音」をMSXのメモリーに読み込み、オーディオ出力端子から出そうと言うわけです。

いくらデジタルオーディオと言っても、A/DやD/Aコンバーターのない MSXですから、その音質は目をつぶることにして、一つの実験として楽しんでみてください。

さて、音という周波数の高い信号を扱うには、遅いBASICではとうてい不可能です。そこで機械語を使用するわけですが、みなさんの中には、機械語を扱ったことがない人も多いと思い、プログラムは一見BASICで作ってあるように見えるように開発しました（**リスト4a**）。

しかし、その実体は**リスト4b**に示すようなアセンブラと呼ばれるもので作られているのです。このアセンブラは、機械語のプログラムを開発するためのもので、みなさんはなじみがない人がほとんどだと思いますが、そのうち必ず役に立ちますのでよく覚えておいてください。

プログラムの使い方

まず**リスト4a** を入力して必ずカセットにセーブしておきます。

DATA

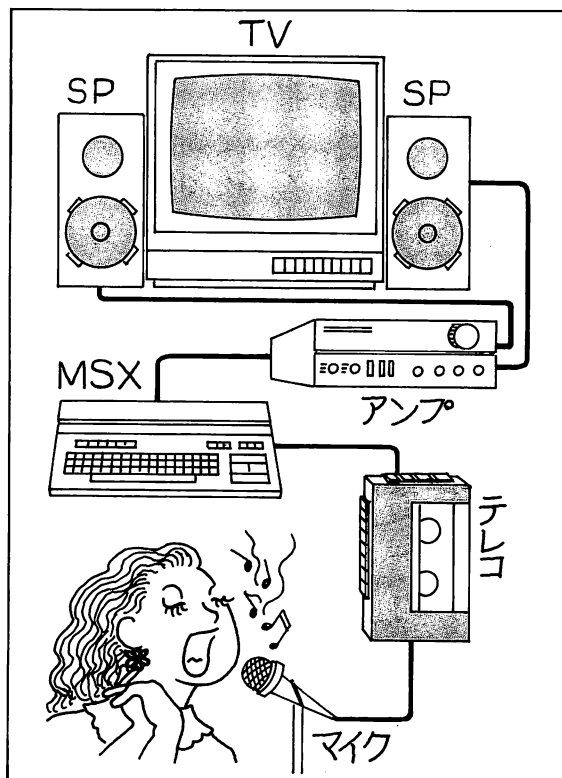
機械語のプログラムですので、**DATA** 文のデータを1文字でも間違いますとCPUが暴走するためです。

RUN

次に **RUN** しますと、メッセージが出て入力待ちとなります。以下に各コマンドの説明をします（なおコマンドは、すべて小文字で入力してください)。

1．t（テスト・コマンド)

tを押すと、カセット入力からの音をMSX内で変換したものをそのままオーディオ出力端子から出します。テレコの頭出しや入力レベルの調整に使用してください。

このコマンドから抜けるには、コントロールスイッチと、ストップスイッチを同時に押します。

2．r（録音コマンド)

rを押すと、まず記録スピードを聞いてきます。1が最も速く、9が一番遅くなります。1の場合約20秒間、9の場合は約30秒間（32Kバイトシステムの場合）記録することができます。数字が大きいほうが記録時間は長いのですが、逆に音質は悪くなります。

スピードを入力した後、何でも良いからキーを押すと記録がスタートします。メモリーいっぱいに音を記録すると自動的にメニューに戻ります。

3．p（再生コマンド)

pを押すと再生スピードを聞いてきます。1が一番速く9が一番遅くなります。rコマンドで記録したスピードより速くすると早回

しに、逆に遅くすると、モゴモゴした感じになります。

再生をやめるには、コントロールスイッチとストップスイッチを同時に押します。

4．o（ループプレイコマンド）

このスイッチを押すたびに画面には、LOOP PLAY とNON-LOOP PLAYが交互に表示されます。

ループプレイとは、ちょうどエンドレスのテープと同じで、pコマンド再生すると何度でも再生します。逆にノンループプレイは、1度再生したらメニューに戻ります。

5．s（セーブカセット）

sを押すと、現在メモリー上にある音のデータを、カセットにセーブします。32Kバイトシステムでは15分ぐらいかかりますので、お茶でも飲んで待つと良いでしょう。

6．ℓ（ロードカセット）

ℓ（エル）を入力すると、カセットから音のデータを読み込みます。セーブの場合と同じ時間がかかります。

このプログラムは16Kバイト以上のシステムで動作し、ディスクがつないである場合ははずしてからプログラムを実行してください。

リスト4a(その1) コンピュレコーダーII

```
10 ' Compu recoder version ][
20 ' by Haruka Takagi
30 ' copyright 1984 (C) NATS
40 MAXFILES=0:CLEAR 10,&HF05F
50 SCREEN 0:CLS
60 PRINT "****************************************";
70 PRINT "*      Compu recorder version ][      *";
80 PRINT "*    by Haruka Takagi                 *";
90 PRINT "*    copyright 1984 (C) NATS          *";
100 PRINT "*    copyright 1984 (C)               *";
110 PRINT "*   Seibundo-Shinkousha publishing    *";
120 PRINT "****************************************";
130 PRINT:PRINT:PRINT
140 PRINT "Please wait for a moment!!!"
150 AD=&HF060
160 READ D$:IF D$="end" THEN GOTO 200
170 D=VAL("&H"+D$)
180 POKE AD,D:AD=AD+1
190 GOTO 160
200 DEF USR=&HF060
210 A=USR(0)
220 'machine language data
230 '&hF060 -> &hF37f
```

リスト4a(その2) コンピュレコーダーII

```
240 '
250 DATA FB,11,D5,F1,CD,96,F0,3A
260 DATA 7A,F3,A7,11,45,F3,20,03
270 DATA 11,60,F3,CD,96,F0,CD,9F
280 DATA 00,FE,72,28,3A,FE,70,28
290 DATA 7C,FE,74,28,1A,FE,73,CA
300 DATA 41,F1,FE,6C,CA,71,F1,FE
310 DATA 6F,CA,C8,F1,18,CA,1A,A7
320 DATA C8,CD,A2,00,13,18,F7,11
330 DATA 70,F2,CD,96,F0,F3,3E,0E
340 DATA CD,96,00,E6,80,CD,35,01
350 DATA CD,B7,00,38,AB,18,EF,11
360 DATA 8D,F2,CD,96,F0,CD,B4,F1
370 DATA 32,79,F3,11,AF,F2,CD,96
380 DATA F0,CD,9F,00,F3,11,60,F0
390 DATA 2A,48,FC,06,08,00,3E,0E
400 DATA CD,96,00,E6,80,F5,CD,35
410 DATA 01,F1,17,CB,16,CD,AB,F1
420 DATA 10,EC,CD,B7,00,DA,60,F0
430 DATA 23,E5,AF,ED,52,7C,B5,E1
440 DATA 20,D9,C3,60,F0,11,7C,F2
450 DATA CD,96,F0,CD,B4,F1,32,79
460 DATA F3,F3,11,60,F0,2A,48,FC
470 DATA 06,08,4E,3E,0E,CD,96,00
480 DATA E6,00,CB,11,1F,F5,CD,35
490 DATA 01,F1,CD,AB,F1,10,EC,CD
500 DATA B7,00,DA,60,F0,23,E5,AF
510 DATA ED,52,7C,B5,E1,C2,10,F1
520 DATA 3A,7A,F3,A7,28,CC,C3,60
530 DATA F0,11,98,F2,CD,96,F0,CD
540 DATA 9F,00,2A,48,FC,E5,3E,FF
550 DATA CD,EA,00,E1,38,49,7E,E5
560 DATA CD,ED,00,E1,38,41,23,E5
570 DATA AF,11,60,F0,ED,52,7C,B5
580 DATA E1,20,EB,CD,F0,00,C3,60
590 DATA F0,11,BD,F2,CD,96,F0,CD
600 DATA 9F,00,2A,48,FC,E5,CD,E1
```

```
610 DATA 00,E1,38,1B,E5,CD,E4,00
620 DATA E1,38,14,77,23,E5,AF,11
630 DATA 60,F0,ED,52,7C,B5,E1,20
640 DATA EB,CD,E7,00,C3,60,F0,11
650 DATA E4,F2,CD,96,F0,CD,9F,00
660 DATA C3,60,F0,F5,3A,79,F3,3D
670 DATA 20,FD,F1,C9,11,14,F3,CD
680 DATA 96,F0,CD,9F,00,FE,31,38
690 DATA F3,FE,3A,30,EF,D6,30,C9
700 DATA 3A,7A,F3,E6,01,EE,01,32
710 DATA 7A,F3,C3,60,F0,0C,43,6F
720 DATA 6D,70,75,2D,72,65,63,6F
730 DATA 64,65,72,0D,0A,56,65,72
740 DATA 73,69,6F,6E,20,5D,5B,0D
750 DATA 0A,62,79,20,48,61,72,75
760 DATA 6B,61,20,54,61,6B,61,67
770 DATA 69,0D,0A,63,6F,70,79,72
780 DATA 69,67,68,74,20,31,39,38
790 DATA 34,20,4E,41,54,53,0D,0A
800 DATA 0D,0A,72,2E,2E,52,45,43
810 DATA 20,6D,6F,64,65,0D,0A,70
820 DATA 2E,2E,50,42,20,20,6D,6F
830 DATA 64,65,0D,0A,74,2E,2E,54
840 DATA 45,53,54,20,6D,6F,64,65
850 DATA 0D,0A,73,2E,2E,53,41,56
860 DATA 45,20,64,61,74,61,0D,0A
870 DATA 6C,2E,2E,4C,4F,41,44,20
880 DATA 64,61,74,61,0D,0A,6F,2E
890 DATA 2E,4C,4F,4F,50,20,70,6C
900 DATA 61,79,20,73,77,0D,0A,00
910 DATA 54,65,73,74,20,6D,6F,64
920 DATA 65,0D,0A,00,50,4C,41,59
930 DATA 20,42,41,43,4B,20,4D,4F
940 DATA 44,45,0D,0A,00,52,45,43
950 DATA 20,4D,4F,44,45,0D,0A,00
960 DATA 53,41,56,45,20,54,4F,20
970 DATA 43,41,53,53,45,54,54,45
980 DATA 20,54,41,50,45,0D,0A,48
990 DATA 49,54,20,41,4E,59,20,4B
```

リスト4a(その4) コンピュレコーダーII

```
1000 DATA 45,59,0D,0A,00,4C,4F,41
1010 DATA 44,20,46,52,4F,4D,20,43
1020 DATA 41,53,53,45,54,54,45,20
1030 DATA 54,41,50,45,0D,0A,48,49
1040 DATA 54,20,41,4E,59,20,4B,45
1050 DATA 59,0D,0A,00,43,41,53,53
1060 DATA 45,54,54,45,20,49,53,20
1070 DATA 45,52,52,4F,52,20,4F,52
1080 DATA 20,41,42,4F,52,54,45,44
1090 DATA 20,21,21,07,0D,0A,48,49
1100 DATA 54,20,41,4E,59,20,4B,45
1110 DATA 59,0D,0A,00,49,4E,50,55
1120 DATA 54,20,52,45,43,2F,50,42
1130 DATA 20,53,50,45,45,44,20,28
1140 DATA 31,7E,39,29,0D,0A,31,2E
1150 DATA 2E,2E,46,41,53,54,0D,0A
1160 DATA 39,2E,2E,2E,53,4C,4F,57
1170 DATA 0D,0A,0D,0A,00,28,6E,6F
1180 DATA 6E,2D,6C,6F,6F,70,20,70
1190 DATA 6C,61,79,20,6D,6F,64,65
1200 DATA 20,6E,6F,77,29,0D,0A,00
1210 DATA 28,6C,6F,6F,70,20,70,6C
1220 DATA 61,79,20,6D,6F,64,65,20
1230 DATA 6E,6F,77,29,0D,0A,0D,0A
1240 DATA 00,00,00,FF,00,00,FF,04
1250 DATA end
```

リスト4b(その1)

```
1000:                       ;Compu-recorder program for "MSX"
1010:                       ;Version II
1020:                       ;by Haruka Takagi
1030:                       ;copyright 1984 NATS
1040:                       ;
1050:   0096 =              RDPSG     EQU       0096H
1060:   0135 =              CHGSND    EQU       0135H
1070:   00B7 =              BREAKX    EQU       00B7H
1080:   FC4B =              BOTTOM    EQU       0FC4BH
1090:   009F =              CHGET     EQU       009FH
1100:   00A2 =              CHPUT     EQU       00A2H
1110:   00E1 =              TAPION    EQU       00E1H
1120:   00E4 =              TAPIN     EQU       00E4H
1130:   00E7 =              TAPIOF    EQU       00E7H
1140:   00EA =              TAPOON    EQU       00EAH
1150:   00ED =              TAPOUT    EQU       00EDH
1160:   00F0 =              TAPOOF    EQU       00F0H
1170:                       ;
1180:   F060                          ORG       0F060H
1190:   F060 FB             RTOP:     EI
1200:   F061 11D5F1                   LD        DE,EMSG
1210:   F064 CD96F0                   CALL      MSG
1220:   F067 3A7AF3                   LD        A,(LFLAG)
1230:   F06A A7                       AND       A
1240:   F06B 1145F3                   LD        DE,NLFMSG
1250:   F06E 2003                     JR        NZ,MMM
1260:   F070 1160F3                   LD        DE,LFMSG
1270:   F073 CD96F0         MMM:      CALL      MSG
1280:   F076 CD9F00                   CALL      CHGET
1290:   F079 FE72                     CP        'r'
1300:   F07B 283A                     JR        Z,REC
1310:   F07D FE70                     CP        'p'
1320:   F07F 287C                     JR        Z,PB
1330:   F081 FE74                     CP        't'
1340:   F083 281A                     JR        Z,TEST
1350:   F085 FE73                     CP        's'
1360:   F087 CA41F1                   JP        Z,SAVE
1370:   F08A FE6C                     CP        'l'
1380:   F08C CA71F1                   JP        Z,LOAD
1390:   F08F FE6F                     CP        'o'
1400:   F091 CACBF1                   JP        Z,LPLAY
1410:   F094 18CA                     JR        RTOP
1420:                       ;
1430:   F096 1A             MSG:      LD        A,(DE)
1440:   F097 A7                       AND       A
1450:   F098 C8                       RET       Z
1460:   F099 CDA200                   CALL      CHPUT
1470:   F09C 13                       INC       DE
1480:   F09D 18F7                     JR        MSG
1490:                       ;
```

リスト4b(その2)

```
1500:   F09F 1170F2         TEST:     LD        DE,TMSG
1510:   F0A2 CD96F0                   CALL      MSG
1520:   F0A5 F3                       DI
1530:   F0A6 3E0E           TEST1:    LD        A,14
1540:   F0A8 CD9600                   CALL      RDPSG
1550:   F0AB E680                     AND       80H
1560:   F0AD CD3501                   CALL      CHGSND
1570:   F0B0 CDB700                   CALL      BREAKX
1580:   F0B3 38AB                     JR        C,RTOP
1590:   F0B5 18EF                     JR        TEST1
1600:                       ;
1610:   F0B7 118DF2         REC:      LD        DE,RMSG
1620:   F0BA CD96F0                   CALL      MSG
1630:   F0BD CDB4F1                   CALL      SPEED
1640:   F0C0 3279F3                   LD        (WAITF),A
1650:   F0C3 11AFF2                   LD        DE,HMSG
1660:   F0C6 CD96F0                   CALL      MSG
1670:   F0C9 CD9F00                   CALL      CHGET
1680:   F0CC F3                       DI
1690:   F0CD 1160F0                   LD        DE,RTOP
1700:   F0D0 2A48FC                   LD        HL,(BOTTOM)
1710:   F0D3 0608           RD2:      LD        B,8
1720:   F0D5 00                       NOP
1730:   F0D6 3E0E           RD1:      LD        A,14
1740:   F0D8 CD9600                   CALL      RDPSG
1750:   F0DB E680                     AND       80H
1760:   F0DD F5                       PUSH      AF
1770:   F0DE CD3501                   CALL      CHGSND
1780:   F0E1 F1                       POP       AF
1790:   F0E2 17                       RLA
1800:   F0E3 CB16                     RL        (HL)
1810:   F0E5 CDABF1                   CALL      WAIT
1820:   F0E8 10EC                     DJNZ      RD1
1830:   F0EA CDB700                   CALL      BREAKX
1840:   F0ED DA60F0                   JP        C,RTOP
1850:   F0F0 23                       INC       HL
1860:   F0F1 E5                       PUSH      HL
1870:   F0F2 AF                       XOR       A
1880:   F0F3 ED52                     SBC       HL,DE
1890:   F0F5 7C                       LD        A,H
1900:   F0F6 B5                       OR        L
1910:   F0F7 E1                       POP       HL
1920:   F0F8 20D9                     JR        NZ,RD2
1930:   F0FA C360F0                   JP        RTOP
1940:                       ;
1950:   F0FD 117CF2         PB:       LD        DE,PMSG
1960:   F100 CD96F0                   CALL      MSG
1970:   F103 CDB4F1                   CALL      SPEED
1980:   F106 3279F3                   LD        (WAITF),A
1990:   F109 F3                       DI
```

リスト4b(その3)

```
2000:  F10A 1160F0   PLP:   LD     DE,RTOP
2010:  F10D 2A48FC          LD     HL,(BOTTOM)
2020:  F110 0608     FB2:   LD     B,8
2030:  F112 4E              LD     C,(HL)
2040:  F113 3E0E     FB1:   LD     A,14
2050:  F115 CD9600          CALL   RDPSG
2060:  F118 E600            AND    00H
2070:  F11A CB11            RL     C
2080:  F11C 1F              RRA
2090:  F11D F5              PUSH   AF
2100:  F11E CD3501          CALL   CHGSND
2110:  F121 F1              POP    AF
2120:  F122 CDABF1          CALL   WAIT
2130:  F125 10EC            DJNZ   FB1
2140:  F127 CDB700          CALL   BREAKX
2150:  F12A DA60F0          JP     C,RTOP
2160:  F12D 23              INC    HL
2170:  F12E E5              PUSH   HL
2180:  F12F AF              XOR    A
2190:  F130 ED52            SBC    HL,DE
2200:  F132 7C              LD     A,H
2210:  F133 B5              OR     L
2220:  F134 E1              POP    HL
2230:  F135 C210F1          JP     NZ,FB2
2240:  F138 3A7AF3          LD     A,(LFLAG)
2250:  F13B A7              AND    A
2260:  F13C 28CC            JR     Z,PLP
2270:  F13E C360F0          JP     RTOP
2280:                ;
2290:  F141 119BF2   SAVE:  LD     DE,SVMSG
2300:  F144 CD96F0          CALL   MSG
2310:  F147 CD9F00          CALL   CHGET
2320:  F14A 2A48FC          LD     HL,(BOTTOM)
2330:  F14D E5              PUSH   HL
2340:  F14E 3EFF            LD     A,0FFH
2350:  F150 CDEA00          CALL   TAPOON
2360:  F153 E1              POP    HL
2370:  F154 3849            JR     C,ABORT
2380:  F156 7E       NSV:   LD     A,(HL)
2390:  F157 E5              PUSH   HL
2400:  F158 CDED00          CALL   TAPOUT
2410:  F15B E1              POP    HL
2420:  F15C 3841            JR     C,ABORT
2430:  F15E 23              INC    HL
2440:  F15F E5              PUSH   HL
2450:  F160 AF              XOR    A
2460:  F161 1160F0          LD     DE,RTOP
2470:  F164 ED52            SBC    HL,DE
2480:  F166 7C              LD     A,H
2490:  F167 B5              OR     L
```

リスト4b(その4)

```
2500:  F168 E1              POP    HL
2510:  F169 20EB            JR     NZ,NSV
2520:  F16B CDF000          CALL   TAPOOF
2530:  F16E C360F0          JP     RTOP
2540:                ;
2550:  F171 11BDF2   LOAD:  LD     DE,LDMSG
2560:  F174 CD96F0          CALL   MSG
2570:  F177 CD9F00          CALL   CHGET
2580:  F17A 2A48FC          LD     HL,(BOTTOM)
2590:  F17D E5              PUSH   HL
2600:  F17E CDE100          CALL   TAPION
2610:  F181 E1              POP    HL
2620:  F182 381B            JR     C,ABORT
2630:  F184 E5       NLD:   PUSH   HL
2640:  F185 CDE400          CALL   TAPIN
2650:  F188 E1              POP    HL
2660:  F189 3814            JR     C,ABORT
2670:  F18B 77              LD     (HL),A
2680:  F18C 23              INC    HL
2690:  F18D E5              PUSH   HL
2700:  F18E AF              XOR    A
2710:  F18F 1160F0          LD     DE,RTOP
2720:  F192 ED52            SBC    HL,DE
2730:  F194 7C              LD     A,H
2740:  F195 B5              OR     L
2750:  F196 E1              POP    HL
2760:  F197 20EB            JR     NZ,NLD
2770:  F199 CDE700          CALL   TAPIOF
2780:  F19C C360F0          JP     RTOP
2790:                ;
2800:  F19F 11E4F2   ABORT: LD     DE,ABMSG
2810:  F1A2 CD96F0          CALL   MSG
2820:  F1A5 CD9F00          CALL   CHGET
2830:  F1A8 C360F0          JP     RTOP
2840:                ;
2850:  F1AB F5       WAIT:  PUSH   AF
2860:  F1AC 3A79F3          LD     A,(WAITF)
2870:  F1AF 3D       LLL:   DEC    A
2880:  F1B0 20FD            JR     NZ,LLL
2890:  F1B2 F1              POP    AF
2900:  F1B3 C9              RET
2910:                ;
2920:  F1B4 1114F3   SPEED: LD     DE,SMSG
2930:  F1B7 CD96F0          CALL   MSG
2940:  F1BA CD9F00          CALL   CHGET
2950:  F1BD FE31            CP     '1'
2960:  F1BF 38F3            JR     C,SPEED
2970:  F1C1 FE3A            CP     ':'
2980:  F1C3 30EF            JR     NC,SPEED
2990:  F1C5 D630            SUB    '0'
```

リスト4b(その5)

```
3000:   F1C7 C9                    RET
3010:                    ;
3020:   F1C8 3A7AF3      LPLAY:    LD      A,(LFLAG)
3030:   F1CB E601                  AND     1
3040:   F1CD EE01                  XOR     1
3050:   F1CF 327AF3                LD      (LFLAG),A
3060:   F1D2 C360F0                JP      RTOP
3070:                    ;
3080:   F1D5 0C          EMSG:     DEFB    0CH
3090:   F1D6 436F6D70              DEFM    'Compu-recoder'
3100:   F1E3 0D0A                  DEFB    0DH,0AH
3110:   F1E5 56657273              DEFM    'Version ][' 
3120:   F1EF 0D0A                  DEFB    0DH,0AH
3130:   F1F1 62792048              DEFM    'by Haruka Takagi'
3140:   F201 0D0A                  DEFB    0DH,0AH
3150:   F203 636F7079              DEFM    'copyright 1984 NATS'
3160:   F216 0D0A0D0A              DEFB    0DH,0AH,0DH,0AH
3170:   F21A 722E2E52              DEFM    'r..REC mode'
3180:   F225 0D0A                  DEFB    0DH,0AH
3190:   F227 702E2E50              DEFM    'p..PB  mode'
3200:   F232 0D0A                  DEFB    0DH,0AH
3210:   F234 742E2E54              DEFM    't..TEST mode'
3220:   F240 0D0A                  DEFB    0DH,0AH
3230:   F242 732E2E53              DEFM    's..SAVE data'
3240:   F24E 0D0A                  DEFB    0DH,0AH
3250:   F250 6C2E2E4C              DEFM    'l..LOAD data'
3260:   F25C 0D0A                  DEFB    0DH,0AH
3270:   F25E 6F2E2E4C              DEFM    'o..LOOP play sw'
3280:   F26D 0D0A00                DEFB    0DH,0AH,00H
3290:   F270 54657374    TMSG:     DEFM    'Test mode'
3300:   F279 0D0A00                DEFB    0DH,0AH,00H
3310:   F27C 504C4159    PMSG:     DEFM    'PLAY BACK MODE'
3320:   F28A 0D0A00                DEFB    0DH,0AH,00H
3330:   F28D 52454320    RMSG:     DEFM    'REC MODE'
3340:   F295 0D0A00                DEFB    0DH,0AH,00H
3350:   F298 53415645    SVMSG:    DEFM    'SAVE TO CASSETTE TAPE'
3360:   F2AD 0D0A                  DEFB    0DH,0AH
3370:   F2AF 48495420    HMSG:     DEFM    'HIT ANY KEY'
3380:   F2BA 0D0A00                DEFB    0DH,0AH,00H
3390:   F2BD 4C4F4144    LDMSG:    DEFM    'LOAD FROM CASSETTE TAPE'
3400:   F2D4 0D0A                  DEFB    0DH,0AH
3410:   F2D6 48495420              DEFM    'HIT ANY KEY'
3420:   F2E1 0D0A00                DEFB    0DH,0AH,00H
3430:   F2E4 43415353    ABMSG:    DEFM    'CASSETTE IS ERROR OR ABORTED !!'
3440:   F303 070D0A                DEFB    07H,0DH,0AH
3450:   F306 48495420              DEFM    'HIT ANY KEY'
3460:   F311 0D0A00                DEFB    0DH,0AH,00H
3470:   F314 494E5055    SMSG:     DEFM    'INPUT REC/PB SPEED (1~9)'
3480:
3490:   F32C 0D0A                  DEFB    0DH,0AH
```

リスト4b(その6)

```
3500:   F32E 312E2E2E              DEFM    '1...FAST'
3510:   F336 0D0A                  DEFB    0DH,0AH
3520:   F338 392E2E2E              DEFM    '9...SLOW'
3530:   F340 0D0A0D0A              DEFB    0DH,0AH,0DH,0AH,00H
3540:   F345 286E6F6E    NLPMSG:   DEFM    '(non-loop play mode now)'
3550:   F35D 0D0A00                DEFB    0DH,0AH,00H
3560:   F360 286C6F6F    LPMSG:    DEFM    '(loop play mode now)'
3570:   F374 0D0A0D0A              DEFB    0DH,0AH,0DH,0AH,00H
3580:                    ;
3590:   F379             WAITF:    DEFS    1
3600:   F37A             LFLAG:    DEFS    1
3610:
3620:   F37B                       END
```

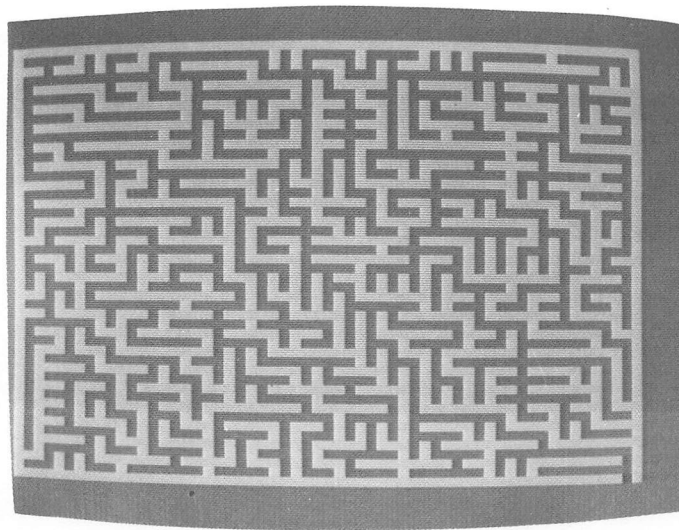
```
F2E4  ABMSG    F19F  ABORT    FC48  BOTTOM   00B7  BREAKX
009F  CHGET    0135  CHGSND   00A2  CHPUT    F1D5  EMSG
F2AF  HMSG     F2BD  LDMSG    F37A  LFLAG    F1AF  LLL
F171  LOAD     F1C8  LPLAY    F360  LPMSG    F073  MMM
F096  MSG      F184  NLD      F345  NLPMSG   F156  NSV
F0FD  PB       F113  PB1      F110  PB2      F10A  PLP
F27C  PMSG     F0D6  RD1      F0D3  RD2      0096  RDPSG
F0B7  REC      F28D  RMSG     F060  RTOP     F141  SAVE
F314  SMSG     F1B4  SPEED    F298  SVMSG    00E4  TAPIN
00E7  TAPIOF   00E1  TAPION   00F0  TAPOOF   00EA  TAPOON
00ED  TAPOUT   F09F  TEST     F0A6  TEST1    F270  TMSG
F1AB  WAIT     F379  WAITF
```

スーパーメイズ

最近、迷路ゲームをあちこちで見かけますが、皆さんは、迷路ゲームは好きですか？　なかなかユニークな迷路がたくさんあっておもしろいのですが、いざ自分で迷路を作ろうとすると、これがなかなかたいへんですね。できた！　と思ってやってみると迷路にならず、いきづまりなんてこともあれば、なあ～んだ、こんなに簡単なの、なんてこともありますよね。

そこで、これをMSXにやらせちゃおうというのが、このゲームです。どうせなら、画面いっぱいに作って、より複雑にした方がおもしろいですね。

では、始めましょう。君は、何秒でぬけることができるでしょうか？

遊び方

まず、**リスト5**を入力してRUNしてください。すると迷路を作り始めます。この迷路作成にはしばらく時間がかかりますので、リストでもながめて一息入れてください。

迷路が完成しますと、ピッと音がしてゲームスタートとなります。左上の赤色の四角があなたで、出口は右下となっています。このゲームでは、本体のカーソルキーとジョイスティックのNo.1を同時に使えますので、どちらでもゲームの途中で変えることができます。うちのグループでジョイスティックで遊んでいた人のところで本体のカーソルキーをいじったために、ひっぱたかれた人がおります。逆に2人で1人はジョイスティック、もう一人は本体のカーソルキーで動かすとおもしろいんじゃないでしょうか。

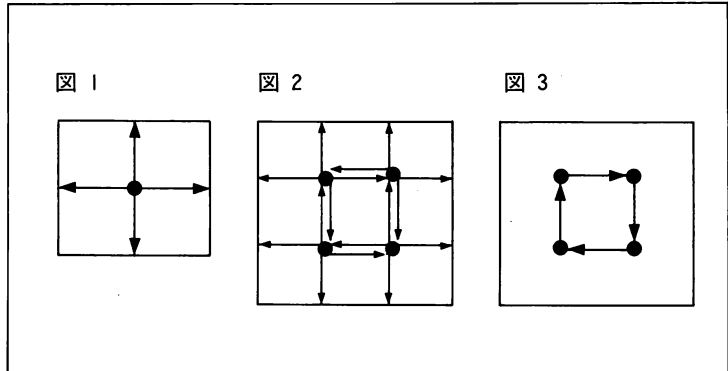
STICK(0)
STICK(1)

みごと迷路を脱出しますと、脱出するのにかかった時間と、その時間に対するキツ～イ評価とへんちくりんな音楽が流れます。このゲームのこつは、ゲームがスタートしても、すぐに動かさず、まず目で出口から逆にスタートまでをたどってからおもむろに動かすと、実にスムーズに抜けることができます。

プログラム

このプログラムの最大のポイントは、何と言っても迷路を作るアルゴリズムです。このアルゴリズムは、本書の「ハイパーチェイサー」のスネークパターン方式ではなく、俗称スキャン方式と呼ばれている方法を使用しています。まずこのアルゴリズムを解説しましょう。

さて、はじめに正方形を書いてその真中に点を打ちます。次にこの点を起点にして上下左右のどちらか一方に線を引きます(**図1**)。これがこの方式の基本形です。次に**図2**のようにこの基本形を4つにしてみます。つまり4つの起点から同じように上下左右に線を1本だけ引くと、だんだん迷路らしくなってきました。しかし注意しないと、**図3**のように道がループを作ってしまうことがあります。そこで、次のような規則を作り、迷路を作成します。

1．壁になる線を引くのは必ず一番上の段の点から始める

2．一番上の段以外の点から壁になる線を引けるのは下右左の3つだけとする

3．線はかさなってはいけない

つまり一番上の段の点をすべて処理してから下の段の処理を行うということと、一番上以外の段では上方向に線を引かないということで、これを守れば**図3**のようなループはできないことになります。

このアルゴリズムの特長は、考え方がちょっと難しいのですが、高速であるということと道のうねりが多いので、直感的に解くことができないということです。実際のプログラムでは270～570行が迷路作成の部分ですので、読んでみてください。ちょっと変更すれば100×100の迷路もなんなく作ることができます。

DRAW

もう一つのこのプログラムの特長は、タイトルに**DRAW**命令を使っていることで、130～190行でMAZE GAMEというきれいなタイトルを作成しています。

POINT

TIME

また、移動の際の壁の判断は、**POINT**命令による色で判断を行っています。時間の表示は、**TIME**変数の値を60で割ることにより近似的に秒数を計算しています。

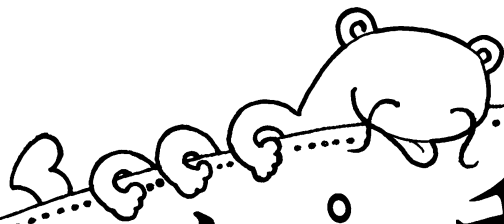
このプログラムは16Kバイト以上のシステムで使用可能で、ディスクシステムでもOKです。

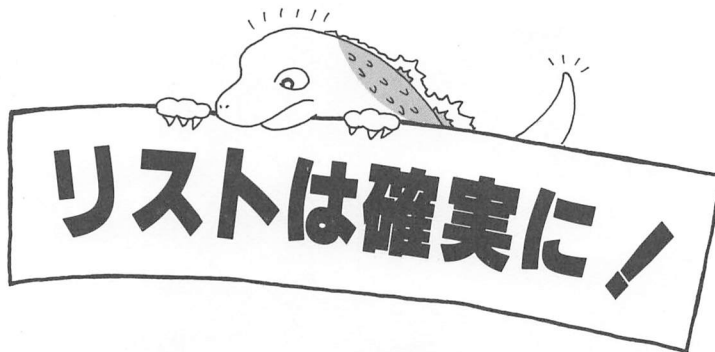
リスト5(その1) スーパーメイズ

```
 10 ' Super maze game
 20 ' by Tamae tama
 30 ' debug and advice by Haruka Takagi
 40 ' copyright 1984 (C) NATS
 50 ' copyright 1984 (C)
 60 '     Seibundo-Shinkousha publishing
 70 '
 80 ' HENSU & SYSTEM INIT
 90 '
100 SCREEN 2:COLOR 15,4,4
110 OPEN "GRP:" FOR OUTPUT AS #1
120 DEFINT A-Z:F=0:R=RND(-TIME)
130 DRAW"BM30,50L20D20R20
140 DRAW"BM40,70U20R20D10L20R10M60,70
150 DRAW"BM70,50R20L10D20
160 DRAW"BM120,70U20M+10,+10M+10,-10D20
170 DRAW"BM150,70U20R20D20U10L20
180 DRAW"BM180,50R20M-20,+20R20
190 DRAW"BM230,50L20D10R10L10D10R20
200 PRESET(50,100):PRINT#1,"By T.Tama & H.Takagi"
210 PRESET(50,130):PRINT#1,"Copyright 1984 NATS
220 PRESET(50,150):PRINT#1,"Copyright 1984 (C)
230 PRESET(50,160):PRINT#1,"Seibundo-Shinkousha
240 PRESET(34,180):PRINT#1,"PUSH TRIGER or SPC KEY !!
250 IF STRIG(0)=-1 OR STRIG(1)=-1 THEN 260 ELSE 250
260 SCREEN 3
```

リスト5(その2) スーパーメイズ

```
270 '
280 ' "WAKU" WRITE
290 '
300 LINE(8,0)-(240,184),15,B
310 PSET(12,4),8
320 PSET(236,184),0
330 '
340 ' "UE EDA" WRITE
350 '
360 FOR X=16 TO 236 STEP 8
370 PSET(X,8),15
380 ON 1+4*RND(1) GOTO 390,410,420,430
390 IF F THEN ON 1+3*RND(1) GOTO 410,420,430
400 PSET(X-4,8),15:GOTO440
410 PSET(4+X,8),15:F=1:GOTO440
420 PSET(X,12),15:F=0:GOTO440
430 PSET(X,4),15:F=0
440 NEXT
450 '
460 ' "NAKA EDA" WRITE
470 '
480 FOR Y=16 TO 180 STEP 8
490 F=0
500 FOR X=16 TO 236 STEP 8
510 PSET(X,Y),15
520 ON 1+3*RND(1) GOTO 530,550,560
530 IF F THEN ON 1+2*RND(1) GOTO 550,560
540 PSET(X-4,Y),15:GOTO 570
550 PSET(4+X,Y),15:F=1:GOTO 570
560 PSET(X,4+Y),15:F=0
570 NEXT:NEXT
```

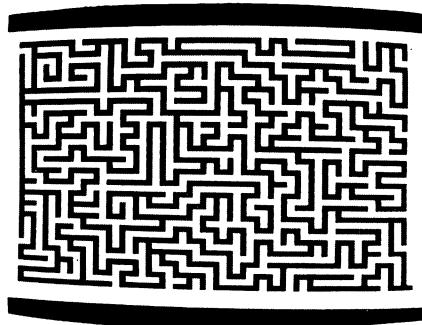
リスト5(その3) スーパーメイズ

```
580 '
590 ' GAME START
600 '
610 X=12:Y=4
620 BEEP:BEEP
630 TIME=0
640 '
650 ' KEY IN
660 '
670 FOR I=0 TO 1
680 J=STICK(I)
690 IF 0=J THEN NEXT:GOTO 670
700 '
710 ' MOVE
720 '
730 IF 3=J AND 15<>POINT(X+4,Y) THEN XX=4:YY=0:GOTO 770
740 IF 7=J AND 15<>POINT(X-4,Y) THEN XX=-4:YY=0:GOTO 77
    0
750 IF 5=J AND 15<>POINT(X,Y+4) THEN XX=0:YY=4:GOTO 770
760 IF 1=J AND 15<>POINT(X,Y-4) THEN XX=0:YY=-4 ELSE 67
    0
770 PRESET(X,Y):X=X+XX:Y=Y+YY
780 PSET (X,Y),8
790 IF X=236 AND Y=184 THEN BEEP:BEEP ELSE FOR I=0 TO 3
    00:NEXT:GOTO670
```

リスト5(その4) スーパーメイズ

```
800 '
810 ' GAME END
820 '
830 T=TIME:FOR I=O TO 2000:NEXT
840 SCREEN 1:KEY OFF
850 LOCATE 9,8:PRINT"GAME OVER"
860 LOCATE 4,10:PRINT"しょよう し゛かん";T¥60;"ヒ゛ョウ"
870 PLAY"O5M1112S10T240EDCDE","O4M1112S10T240ACEACE"
880 PLAY"O5M1112S10T240EDCDE","O4M1112S10T240ACEACE"
890 PLAY"O5M1112S10T240EDCEDCCDEFO6C","O4M1112S10T240AB
    CABCEFG"
900 IF PLAY(0) THEN 900
910 QA=T¥60
920 IF QA<=30 THEN PRINT "    すこ゛い にんけ゛んし゛ゃ ない!!!":GOTO
     980
930 IF QA<=45 THEN PRINT "    たいした もので゛す よ ホント!!!":GOTO
     980
940 IF QA<=60 THEN PRINT "    まあ まあ て゛すね たた゛のひと  ":GOTO
     980
950 IF QA<=90 THEN PRINT "    あなた,にんけ゛ん し゛ゃないよ!!!":GOTO
     980
960 IF QA<=135THEN PRINT "    ノロマ!! ト゛シ゛!! カメ!!  ":GOTO
     980
970 PRINT "    もう,にんけ゛ん やめたほうか゛ いい"
980 LOCATE 0,15:PRINT "     Try again? (y/n)"
990 A$=INKEY$
1000 IF A$="Y" OR A$="y" THEN RUN
1010 IF A$="N" OR A$="n" THEN END ELSE GOTO 990
```

毎日、夜遅くまでマイコンの前に座り込んでいる僕は何をやってるかって、なんてことはない。いっしょうけんめいゲームを作っているんだ。おもしろいゲームを目ざし、どこかのプロコン（プログラムコンテスト）に入賞する夢を見ている。

時は真夜中12時。どこからか時計のボーン、ボーンという音が聞こえてくる。やっと完成間近のプログラムを、テストランさせようと、ファンクションキーを押した。と、その瞬間、CRT 画面がスパークし、気がつくとまったく見知らぬところにいる。あれ～! 僕はたしかプログラムを作っていて、テストランを始めたはずだが…。

な～んて話が始まるんだけど、ようするに君は、プログラムの迷路に入り込んじゃった。よくみると、出口はある。しかし、何かこちらに向かってくるぞー。あ～! バグ虫だ。そこで君に指示する。このバグ虫に捕まらないように、この分岐あり、いきづまりありのプログラム迷路から脱出して欲しい。バグ虫は壁をつぎつぎに破壊してくる。さて君は何秒で脱出できるか。

遊び方

このゲームは画面いっぱいに作られた迷路を、バグ虫につかまらないように右下の出口から脱出するものです。

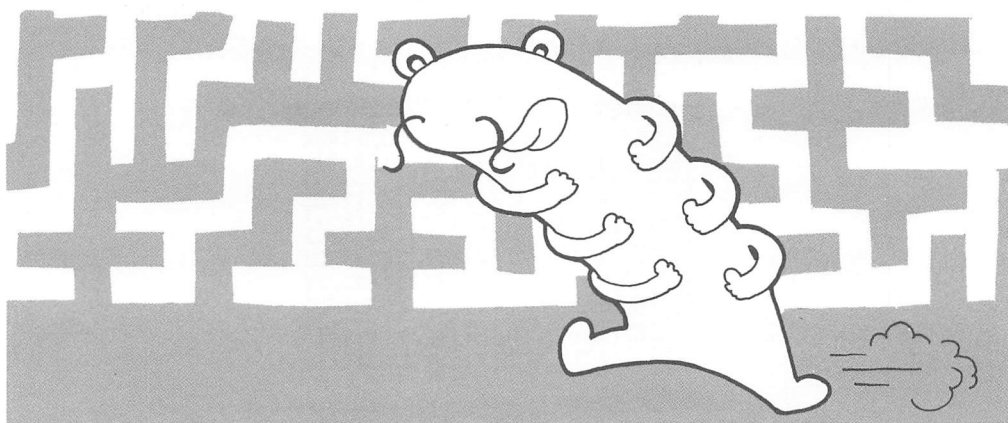
内容はいたって簡単なのですが、バグ虫の追跡をうまくふり切らないと簡単につかまってしまうぞ～。コツとしては、バグ虫をうまく引き付け、逆にバグ虫の破壊した壁の道を通って脱出することなんだ。とはいっても簡単に脱出するのはたいへん難しいんですが。

RUN
INKEY $
RND (1)

まず、**リスト6**を入力し、RUNするとキー入力待ちとなりますので、何でも良いからキーを押してください。この入力待ちになるのは、BASICの乱数の初期値をよくかきまぜるためのもので、これがないと電源入力後は毎回同じ迷路が作られてしまいます。

次に迷路を作りはじめますが、これに少々時間がかかりますのでじっと画面を見て脱出路を考えるなり、お茶でもいっぱい飲むなりして待っていてください。迷路が書き終わると音楽が鳴り、ゲームスタートとなります。

左上で点滅しているのがあなたで、バグ虫は右下（出口）から出てきます。移動は本体のカーソルキーでもジョイスティック（ただしNo.1に接続してください）のどちらでもけっこうです。

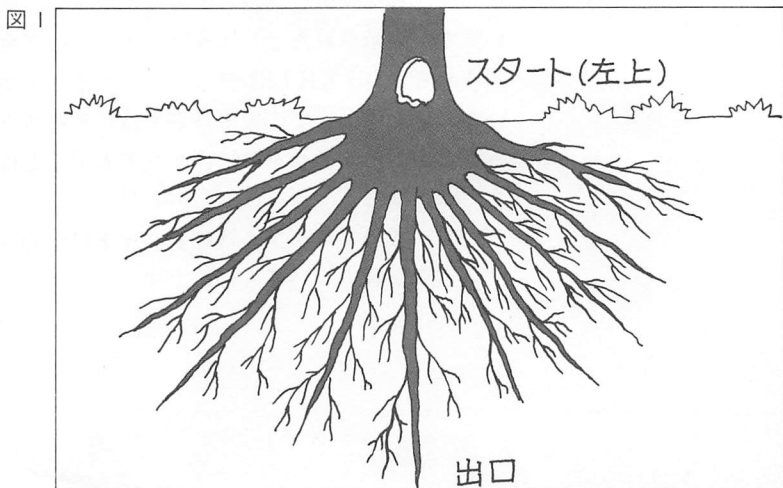
バグ虫からのがれ、うまく脱出できれば、所用時間が表示されます。逆にバグ虫につかまりますと、その時点でゲームオーバーとなってしまうので、慎重に!!

プログラム

SCREEN3

このプログラムは、他のゲームと異なりマルチカラーモード(**SCREEN 3**)を使用しています。このモードは、64×48ドットで低解像度のグラフィックとも言え、ドットが大きいのが特長ですが、逆にドットごとに色を変えることが可能です。迷路を作るには最適なわけです。

さて、このプログラムの中心は何と言っても、迷路の作成フローです。迷路の作り方は何種類もあり、この方法は、迷路を作って行く様がヘビに似ているのでスネークパターンなどとも呼ばれています。つまり迷路の左上の始点から乱数で方向を決定しその方向に壁または、他の道につきあたり、動けなくなるまで道をのばしていく方法です。

この方法を図式化すると**図1**のようになります。この図からわかるように、この方式は迷路が枝状に分かれるため複雑なものとなり、かつスタートから出口までの道が1つしかないという点に特長があります。このアルゴリズムは、方向決定および作成を320〜440行で、また壁の判断を480行で行っています。迷路の作り方は本書のスーパーメイズで違った方法を使用していますので、そちらもあわせて見てください。

図1

以上で迷路が完成し、次はゲームスタートとなるわけです。ゲーム中のポイントは「追いかけ」のプログラムで、ここでは人間とバグ虫の座標を最短とするようにバグ虫を動かしています。つまり、人間の座標を（X, Y)、バグ虫の座標を（A, B）とすると、

$$|(X-A)|+|(Y-B)|$$

すなわち、ABS (X－A) ＋ABS (Y－B)が最小となるように、バグ虫のX軸、Y軸方向の座標を決めているわけです。これはプログラムでは1070～1130で行っています。

この部分はフィードバックを利用しているので、プログラムを追って行くとちょっとわかりにくい点があるかもしれません。また、バグ虫は壁があるとその壁をぶち破って通って行くわけですが、そのときに音を出すようにしています。壁かどうかの判断は1140行を見てもらえばわかるように、**POINT** 命令を使って、そこの色が壁の色かどうかで壁の判断をしています。

POINT

さて、SCREEN 3 モードでは、画面にテキストを表示するとおもしろいことになります。たとえば、

```
10  SCREEN 3 : CLS : OPEN "GRP : " FOR OUTPUT
    AS #1
20  PRINT  #1, "MAZE"
30  GOTO  30
```

というプログラムを実行してみてください。画面には、えらく大きい文字が表示されたでしょう？　実は、バグ虫に捕まったときに表示される YOU KILLED!!　という大きな文字は、これを利用しているわけです。逆にこのモードでは、テキストモード時のような小さな文字は出せませんので、スコア表示などは、ちょっとムリのようです。

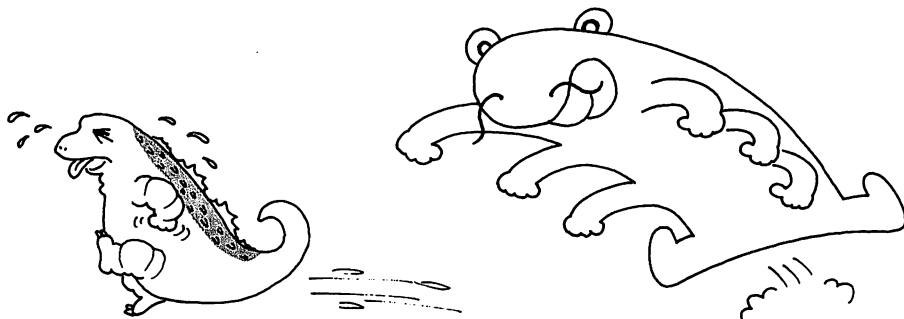
なお、このプログラムは16Kバイト以上のシステムで使用可能でディスクが付いていても OK です。

リスト6(その1) ハイパーチェイサー

```
10 ' Highper chaser
20 ' by Mika Tada
30 ' debug and advice by Haruka Takagi
40 ' copyright 1984 (C) NATS
50 ' copyright 1984 (C) Seibundo-Shinkousha publishing
60 DEFINT A-Z:OPEN "GRP:" FOR OUTPUT AS #1
70 DIM M(31,23):KEY OFF
80 SCREEN 1:COLOR 15,0,0:CLS
90 PRINT "******************************";
100 PRINT "*        Highper chaser      *";
110 PRINT "*                            *";
120 PRINT "*    by M.Tada & H.Takagi    *";
130 PRINT "*    copyright 1984 (C) NATS*";
140 PRINT "*    copyright 1984 (C)      *";
150 PRINT "*    Seibundo-Shinkousha     *";
160 PRINT "******************************";
170 PRINT :PRINT
180 PRINT "       Hit any key to start"
190 IF INKEY$="" THEN A=RND(1):GOTO 190
200 SCREEN 3,0,0:COLOR 15,0,0
210 LINE (0,0)-(255,191),3,BF
220 FOR X=0 TO 31:M(X,0)=1:M(X,23)=1:NEXT
230 FOR Y=0 TO 23:M(0,Y)=1:M(31,Y)=1:NEXT
240 PSET (8,4),1:PSET (244,176),1:MX=240:MY=176
250 M(1,1)=1:M(30,22)=0
260 FOR LX=1 TO 30:FOR LY=1 TO 22
270 X=LX*8:Y=LY*8:X1=LX:Y1=LY
280 IF M(X1,Y1-1)+M(X1,Y1+1)+M(X1-1,Y1)+M(X1+1,Y1)=4 TH
    EN NEXT :NEXT :GOTO 490
290 IF M(X1,Y1)=0 THEN NEXT :NEXT :GOTO 490
300 GOSUB 320
310 NEXT :NEXT :GOTO 490
```

リスト6(その2) ハイパーチェイサー

```
320 '
330 '     MEIROKAKU SUBROUTINE !
340 '
350 ON 1+4*RND(1) GOTO 360,380,400,420
360 IF M(X1+1,Y1) THEN 320
370 LINE (X,Y)-(X+8,Y),1,BF:M(X1+1,Y1)=1:X=X+8:GOTO 440
380 IF M(X1-1,Y1) THEN 320
390 LINE (X,Y)-(X-8,Y),1,BF:M(X1-1,Y1)=1:X=X-8:GOTO 440
400 IF M(X1,Y1+1) THEN 320
410 LINE (X,Y)-(X,Y+8),1,BF:M(X1,Y1+1)=1:Y=Y+8:GOTO 440
420 IF M(X1,Y1-1) THEN 320
430 LINE (X,Y)-(X,Y-8),1,BF:M(X1,Y1-1)=1:Y=Y-8
440 X1=X/8:Y1=Y/8:'POINTER SET
450 '
460 '     MAWARI WA KABE KA ?
470 '
480 IF M(X1,Y1-1)+M(X1,Y1+1)+M(X1-1,Y1)+M(X1+1,Y1)=4 TH
    EN RETURN ELSE 320
```

```
490 '
500 '   MEIRO DASHUTSU !.
510 '
520 D1$="BAGFEDCO6BAGFEDCO5GAGFEDC"
530 D2$="RBAGFEDCO6BAGFEDCO5GAGFED"
540 D3$="RRBAGFEDCO6BAGFEDCO5GAGFE"
550 Z$="O7L32T200V14"
560 PLAY Z$,Z$,Z$
570 PLAY D1$,D2$,D3$
580 X=8:Y=4:DM=0:PSET (8,4),4
590 LP=0:KL=0
600 IF STICK(0)=5 OR STICK(1)=5 THEN TIME=0:GOTO 610 EL
    SE PSET (8,4),DM:DM=DM XOR 4:GOTO 600
610 PRESET (X,Y):Y=Y+4:PSET (X,Y),4 :LOOP=0
620 IF STICK(0)=1 OR STICK(1)=1 THEN 690
630 IF STICK(0)=5 OR STICK(1)=5 THEN 750
640 IF STICK(0)=3 OR STICK(1)=3 THEN 800
650 IF STICK(0)=7 OR STICK(1)=7 THEN 850
660 LP=LP+1:IF LP=3 THEN LP=0
670 IF LP<>0 THEN 620
680 GOSUB 1060:IF KL=1 THEN GOTO 1040 ELSE 620
```

リスト6(その4) ハイパーチェイサー

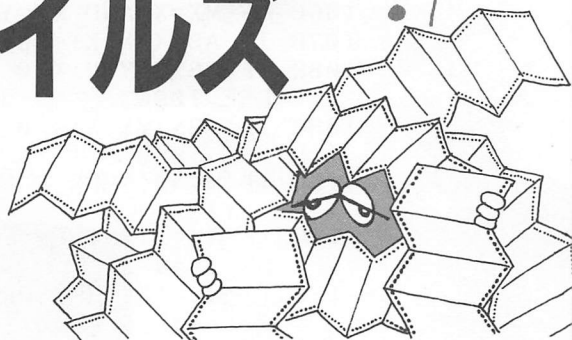
```
690 '
700 '    UP KEY ON !
710 '
720 IF Y<8 THEN 620
730 IF POINT(X,Y-4)=3 THEN 620
740 PRESET (X,Y):Y=Y-4:GOTO 900
750 '
760 '    DOWN KEY ON !
770 '
780 IF POINT(X,Y+4)=3 THEN 620
790 PRESET (X,Y):Y=Y+4:GOTO 900
800 '
810 '    RIGHT KEY ON !
820 '
830 IF POINT(X+4,Y)=3 THEN 620
840 PRESET (X,Y):X=X+4:GOTO 900
850 '
860 '    LEFT KEY ON !
870 '
880 IF POINT(X-4,Y)=3 THEN 620
890 PRESET (X,Y):X=X-4
```

リスト6（その5）ハイパーチェイサー

```
900 PSET (X,Y),4
910 IF KL=1 THEN 1040
920 IF X>243 AND Y>175 THEN 930 ELSE 660
930 BEEP:BEEP:BEEP
940 TM=TIME:FOR X=0 TO 31:FOR Y=0 TO 23:M(X,Y)=0:NEXT :
    NEXT
950 SCREEN 0:CLS
960 FOR DM=0 TO 30
970 LOCATE 0,10:PRINT "      Congratulation!!!!"
980 LOCATE 0,10:PRINT "                        "
990 NEXT
1000 PRINT USING "####.## SEC ｶｶﾘﾏｼﾀ !! ";TM/60:PLAY "C
    ","G","E":PLAY "C16D16E16":PLAY "C","G","E"
1010 PRINT :PRINT "TRY AGAIN ( Y OR N ) "
1020 A$=INKEY$
1030 IF A$="Y" OR A$="y" THEN ERASE M:DIM M(31,23):GOTO
     80 ELSE IF A$="N" OR A$="n" THEN 1050 ELSE 1020
1040 CLS:PSET (0,0):PRINT #1," YOU     KILLED     !!":P
    LAY "G16G+16G16.G+16.G32G+32.":FOR TM=0 TO 5000:NEX
    T:SCREEN 0:GOTO 1010
1050 END
```

リスト6(その6) ハイパーチェイサー

```
1060 IF MX=X AND MY=Y THEN KL=1:RETURN
1070 IF ABS(MX-X)<ABS(MY-Y)*1.18 THEN 1080ELSE 1090
1080 IF ABS(MY-Y)<>0 THEN IF MY>Y THEN 1100 ELSE 1110 E
     LSE 1090
1090 IF ABS(MX-X)<>0 THEN IF MX>X THEN 1120 ELSE 1130 E
     LSE 1080
1100 IF MY>8 THEN PRESET (MX,MY):MY=MY-4:GOTO 1140 ELSE
      1140
1110 IF MY<172 THEN PRESET (MX,MY):MY=MY+4:GOTO 1140 EL
     SE 1140
1120 IF MX>12 THEN PRESET (MX,MY):MX=MX-4:GOTO 1140 ELS
     E 1140
1130 IF MX< 240 THEN PRESET (MX,MY):MX=MX+4:GOTO 1140 E
     LSE 1140
1140 IF POINT (MX,MY)=0 OR POINT (MX,MY)=1 THEN GOTO 11
     50 ELSE PLAY"t255l64ca"
1150 PSET (MX,MY),8
1160 IF PLAY(0) GOTO 1160
1170 IF X=MX AND Y=MY THEN KL=1 ELSE KL=0
1180 RETURN
```

カセットファイルズプログラム

カセットテープというものは便利なもので、1本のカセットに何本もプログラムを入れておくことができますが、あんまり入れすぎたり、カセットの本数が増えてくると、どうしても中にいったい何が入っているのかわからなくなってしまう場合があります。また、ついうっかり内容を書き移しておくのを忘れてそれっきりになってしまったり……。

こんなときは、いちいちカセットからプログラムをロードして内容をチェックしていく方法がありますが、それは簡単には行きません。というのは、MSX では BASIC プログラム(**CSAVE** したもの)、機械語プログラム(**BSAVE** したもの) それに ASCII 形式のプログラムおよびデータ (**SAVE "CAS :**など) と3種類の形式があり、それぞれ対応するロードのし方をしないと、エラーとなってしまいます。　このプログラムは、このようなわずらわしい作業をしなくても、1本のカセットテープに入っているすべての内容（ファイルと呼びます）を表示してくれるもので、表示される内容は、

1．ファイルの形式

BASIC か、機械語か ASCII 形式のプログラムまたはデータかを表示

2．ファイル名

カセットにセーブしたときのファイル名です

3．機械語プログラムのパラメータ

機械語プログラムの場合は、さらにプログラムのスタート番地、終了番地、実行番地を表示します

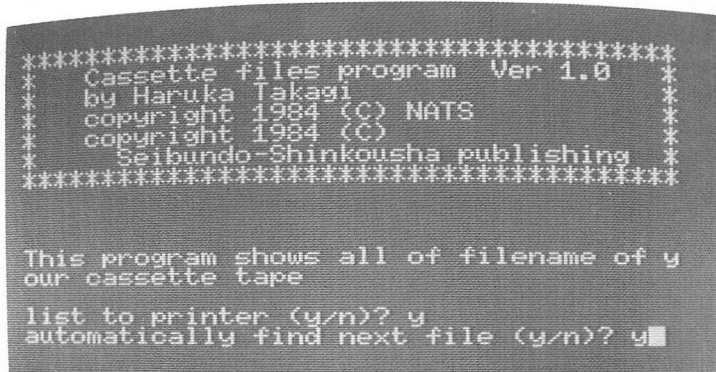
というように、そのファイルの内容を知るのに必要な情報を表示してくれます。

もし、内容まで知りたい場合、この情報をもとに各プログラムに適したコマンドでロードし、その内容を見れば良いわけです。

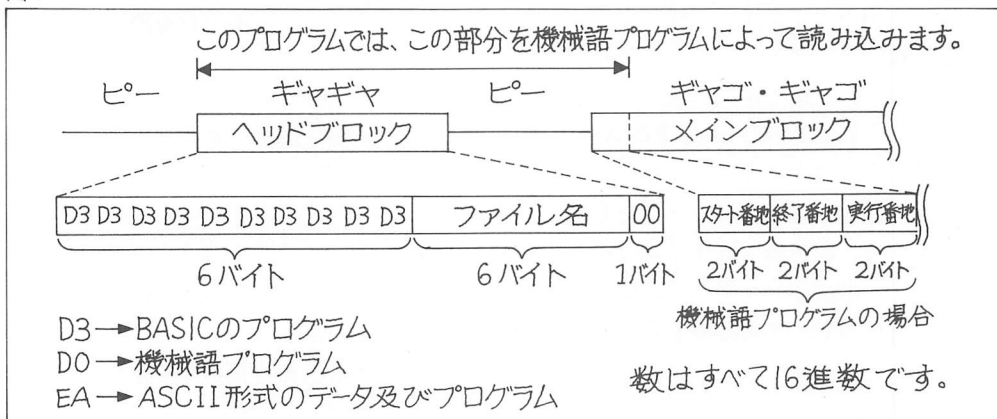
またこの内容は、プリンターに出すこともできますので、プリンターを接続しておけばモニターTVの前にいなくても、ほうっておけば1本のカセットの内容すべてをプリンターに出力してくれます。あとはプリンター用紙を見ながらカセットラベルに内容を書き込んで行けば良いわけです。

```
****************************************
*   Cassette files program  Ver 1.0    *
*   by Haruka Takagi                   *
*   copyright 1984 (C) NATS            *
*   copyright 1984 (C)                 *
*     Seibundo-Shinkousha publishing   *
****************************************


This program shows all of filename of y
our cassette tape

list to printer (y/n)? y
automatically find next file (y/n)? y
```

プログラムの原理

みなさん、一度はカセットにセーブされたプログラムの音を聴いたことがあると思いますが、まったくギャゴ・ギャゴとそうぞうしいね。ところで、この音を注意深く聴くと**図1**のようになっていることに気付くと思います。この最初のギャというところは実はヘッ

図1

ドブロックと呼び、プログラムのファイル名が入っているのです。

この後のピーの後のギャゴ・ギャゴと長く続く音は、プログラムの本体で、特に機械語プログラムの場合は、このギャゴ・ギャゴの頭にプログラムのスタート番地、終了番地、実行番地が入っています。また、プログラムの形式の区別はヘッドブロックの先頭10バイトのデータから判断できます。

さて、カセットにどのような形でセーブされているのかはわかりましたので、後は各ファイルのヘッドブロックとメインブロックの先頭を読み込んで表示してやればプログラムが完成します。

しかし、カセットからのデータの読み込みだけは、とてもとてもBASICでは追い付くことができません。そこで、カセットからのデータの読み込みだけは機械語でやることにしました。

プログラム

プログラムは機械語プログラム(**リスト7b**)により、メモリーのE000番地以後に、各ファイルのヘッドブロックおよびメインブロックの先頭6バイト、計23バイト分をまず読み込み、次にBASICプログラムにより各表示を行っています。特に難しい命令は使用していませんので内容については**リスト7a**を参照してください。

なお機械語プログラムは、BASICプログラム中の670行以後にデータとして格納されており、BASICプログラムで、自動的にメモリーに作られるようになっています。よって、**リスト7a**のみを入力すればけっこうです。

プログラムの使い方

まず**リスト7a**を入力して、セーブしておきます。特に機械語プログラムを併用していますので、入力ミスがありますと暴走する可能性がありますから、必ずセーブしてください。

LPRINT

次にRUNしますと、タイトルが出て、プリンターに出力するかどうかを聞いてきますので、プリンターに出す場合はy、出さない場合はnを押します。そして、オートマチックモードにするかどうかを聞いてきます。

オートマッチックモードとは、１つのファイルの表示を行った後、自動的に次のファイルを連続して調べて行くモードで、プリンターと併用することで、何もしなくても１本のカセット中のすべてのファイルを調べることができます。オートマチックモードではない場合は、１つのファイルの表示をするごとにストップし、次のファイルを調べるか、プログラムを終了するか聞いてきます。

オートマチックモードにする場合はｙ、そうでない場合はｎを入力してください。これでプログラムが起動しますので、後はカセットテープを再生すれば、画面またはプリンターに各ファイルの情報が出力されます。このプログラムは、16Kバイトシステム以上で使用可能で、ディスクがある場合は、ディスクをはずしてください。

使用例

```
file is BASIC program
filename=Cafp1

file is ASCII data or program
filename=Cafp2

filename=TO300:

filename=,1:PRI

filename=";
20

filename=JS=0:G

file is machine language
filename=CafpM
start address=C700
end   address=F37F
entry address=C700
```

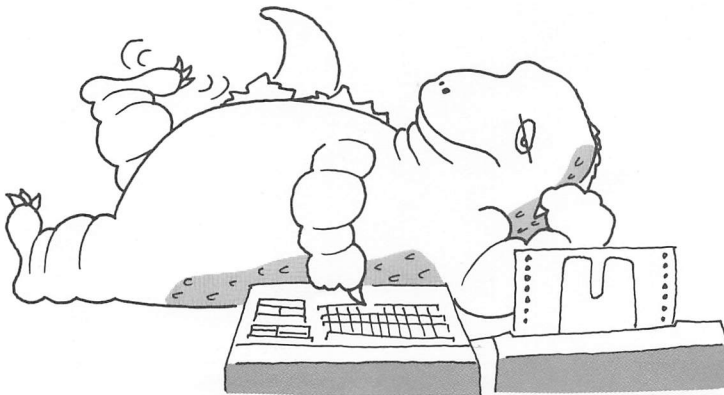
リスト7a(その1) カセットファイルズ

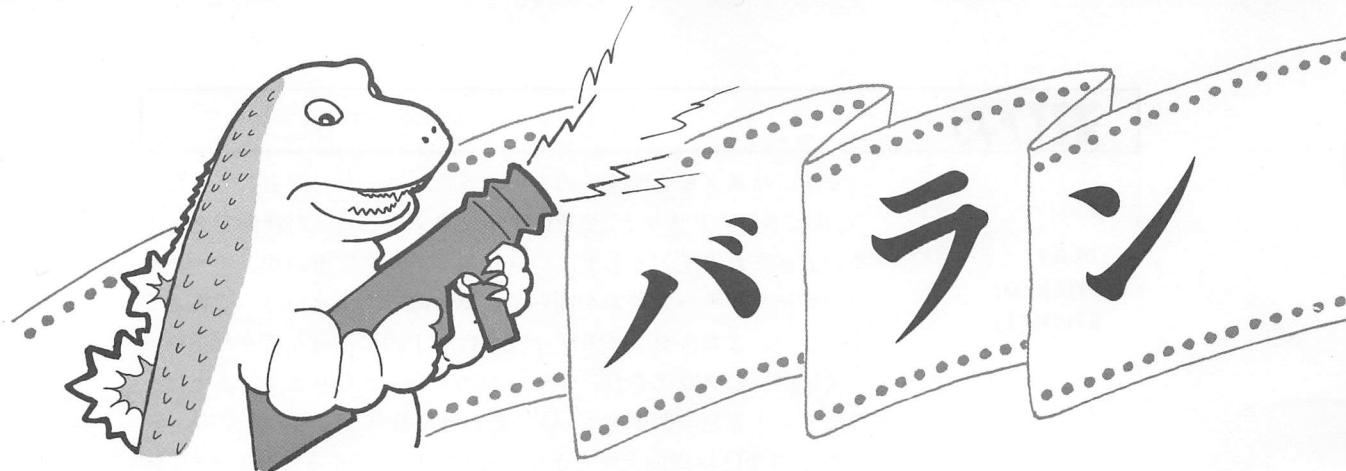
```
10 ' Cassette files program
20 ' by Haruka Takagi
30 ' copyright 1984 (C) NATS
40 CLEAR 100,&HDFFF
50 MAXFILES=0
60 SCREEN 0:CLS
70 PRINT "****************************************";
80 PRINT "*   Cassette files program  Ver 1.0   *";
90 PRINT "*   by Haruka Takagi                  *";
100 PRINT "*   copyright 1984 (C) NATS           *";
110 PRINT "*   copyright 1984 (C)                *";
120 PRINT "*     Seibundo-Shinkousha publishing  *";
130 PRINT "****************************************";
140 PRINT:PRINT:PRINT
150 PRINT "This program shows all of filename of your c
    assette tape"
160 PRINT
170 INPUT "list to printer (y/n)";DM$
180 IF DM$="y" THEN LP=1 ELSE LP=0
190 INPUT "automatically find next file (y/n)";DM$
200 IF DM$="n" THEN AF=0 ELSE AF=1
210 'main routine
220 CLS
230 AD=&HF000
240 READ D$:IF D$="end" THEN GOTO 270
250 D=VAL("&H"+D$):POKE AD,D
260 AD=AD+1:GOTO 240
270 DEF USR=&HF000
280 AD=&HE000
290 FOR I=0 TO &H16
300 POKE (AD+I),0
310 NEXT
320 PRINT "please tape start"
330 A=USR(0)
```

リスト7a（その2）カセットファイルズ

```
340 'error detect
350 D=PEEK(&HF024):IF D<>0 THEN PRINT "tape read error
    !! ":INPUT "retry (y) or end (n)";DM$:IF DM$<>"n" T
    HEN PRINT:GOTO 270 ELSE END
360 'build message data
370 AT=PEEK(&HE000)
380 AD=&HE00A
390 F$=""
400 D=PEEK(AD):AD=AD+1:IF AD=&HE011 THEN GOTO 420
410 F$=F$+CHR$(D):GOTO 400
420 ST=PEEK(AD)+PEEK(AD+1)*256
430 ST$=RIGHT$("000"+HEX$(ST),4)
440 AD=AD+2
450 EN=PEEK(AD)+PEEK(AD+1)*256
460 EN$=RIGHT$("000"+HEX$(EN),4)
470 AD=AD+2
480 SS=PEEK(AD)+PEEK(AD+1)*256
490 SS$=RIGHT$("000"+HEX$(SS),4)
500 'print message
510 PRINT
520 IF AT=&HEA THEN PRINT "file is ASCII data or progra
    m"
530 IF AT=&HD0 THEN PRINT "file is machine language"
540 IF AT=&HD3 THEN PRINT "file is BASIC program"
550 PRINT "filename=";F$
560 IF AT<>&HD0 THEN GOTO 610
570 PRINT "start address=";ST$
580 PRINT "end   address=";EN$
590 PRINT "entry address=";SS$
```

```
600 'print to printer
610 IF LP=0 THEN GOTO 700 ELSE LPRINT
620 IF AT=&HEA THEN LPRINT "file is ASCII data or progr
    am"
630 IF AT=&HD0 THEN LPRINT "file is machine language"
640 IF AT=&HD3 THEN LPRINT "file is BASIC program"
650 LPRINT "filename=";F$
660 IF AT<>&HD0 THEN GOTO 700
670 LPRINT "start address=";ST$
680 LPRINT "end   address=";EN$
690 LPRINT "entry address=";SS$
700 IF AF=1 THEN GOTO 270 ELSE PRINT :INPUT "next (n) o
    r end (e)";DM$
710 IF DM$<>"e" THEN GOTO 270 ELSE END
720 ' machine language data
730 DATA 3E,FF,32,24,F0,CD,E1,00
740 DATA 38,16,21,00,E0,06,17,E5
750 DATA C5,CD,E4,00,C1,E1,38,08
760 DATA 77,23,10,F3,AF,32,24,F0
770 DATA CD,E7,00,C9,end
```

リスト7b カセットファイルズ

```
1000:                    ;Cassette files program
1010:                    ;by Haruka Takagi
1020:                    ;copyright 1984 NATS
1030:                    ;
1040:   00E1 =           TAPION   EQU    00E1H
1050:   00E4 =           TAPIN    EQU    00E4H
1060:   00E7 =           TAPIOF   EQU    00E7H
1070:   E000 =           BUFF     EQU    0E000H
1080:                    ;
1090:   F000                      ORG    0F000H
1100:   F000 3EFF        START:   LD     A,0FFH
1110:   F002 3224F0               LD     (EFLAG),A
1120:   F005 CDE100               CALL   TAPION
1130:   F008 3816                 JR     C,EXIT
1140:   F00A 2100E0               LD     HL,BUFF
1150:   F00D 0617                 LD     B,17H
1160:   F00F E5          NEXT:    PUSH   HL
1170:   F010 C5                   PUSH   BC
1180:   F011 CDE400               CALL   TAPIN
1190:   F014 C1                   POP    BC
1200:   F015 E1                   POP    HL
1210:   F016 3808                 JR     C,EXIT
1220:   F018 77                   LD     (HL),A
1230:   F019 23                   INC    HL
1240:   F01A 10F3                 DJNZ   NEXT
1250:   F01C AF                   XOR    A
1260:   F01D 3224F0               LD     (EFLAG),A
1270:   F020 CDE700      EXIT:    CALL   TAPIOF
1280:   F023 C9                   RET
1290:   F024             EFLAG:   DEFS   1
1300:   F025                      END


E000   BUFF      F024   EFLAG     F020   EXIT      F00F   NEXT
F000   START     00E4   TAPIN     00E7   TAPIOF    00E1   TAPION
```

西暦3001年、人類はある惑星からの攻撃を受けていた。そのころ地球は？というと、バルカン砲をつけただけという単純な宇宙船しかなかったのだ!!　その惑星の攻撃とは、UFOがエイリアンの卵を宇宙船の近くまで運びふ化させるもので、ふ化したエイリアンが宇宙船にとり付くと、なんと一瞬にして宇宙船を破壊してしまうというオソロシーイものなのだ!!

さあ、君は地球を救うために宇宙船バランに乗り込むのだ、ナーンチャッテ。

遊び方

PLAY
STICK (0)
STICK (1)

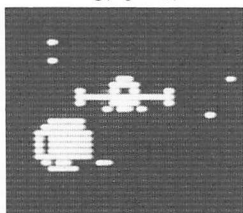
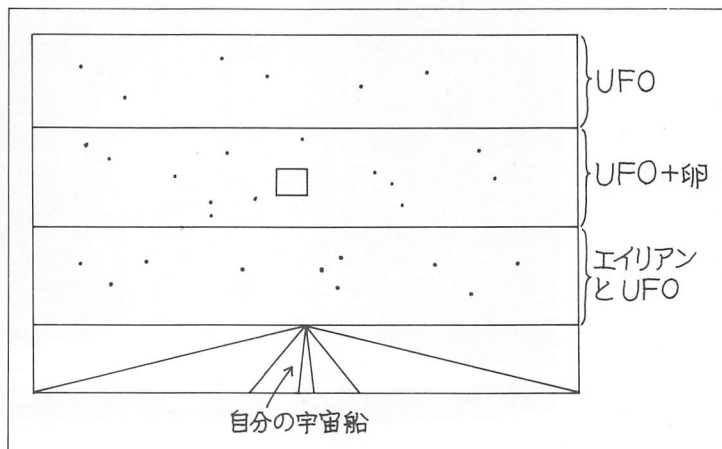
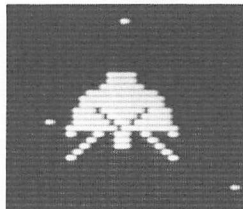
まず、**リスト8**を間違いのないようにシコシコと入力してください。次にRUNしますと、まさに宇宙サウンドと言った趣向でタイトルが表示されます。そして、ジョイスティックで遊ぶか、キーボードのカーソルキーで遊ぶか聞いてきますので、1または2で答えてください。なお、ジョイスティックは№1側に接続してください。

次に画面がクリアされ、ゲームがスタートとなります。あとは、ひたすら十字型の照準をUFOに合わせて打ちまくるのみです。

さて、UFOは画面上から下にだんだん降りてくるわけですが、図のように下に降りるにつれ卵が出現し、さらに降りると卵がふ化し、エイリアンとなります。UFOが卵をだいている(?)状態では、卵を打っても卵はすぐに再生されてしまいますので、UFOを攻撃するようにしてください。逆にUFOを破壊しますと、卵も壊れます。

また卵がふ化してエイリアンになっても、エイリアンはしばらくの間、UFOにくっついています。

この状態ではエイリアンはUFOのバリアに守られていて、エイリアンを破壊することはできませんので、同じようにUFOを攻撃するようにしてください。さて、エイリアンがUFOから離れますと、うろうろとバランに取り付こうとしますので、必死になってエイリアンを攻撃してください。こいつに取り付かれると、一瞬にし

UFOの卵

エイリアン

てバランはクラッシュし、地球は占領されてしまいます。

ところで、UFO をどんどん攻撃して行きますと、最大 6 匹まで増えます。この状態で一定時間が経過しますと、ハデなサウンドと共に WARN!! （注意!!）という文字が表示されます。この状態になると、なんと UFO がどの高さにあってもすべてエイリアンに変化してバランを攻撃してきます。これを必死になって撃ち落とすと、再び UFO が一匹から始まります。

プログラム

プログラムは、けっこう大きなものですが、そのひとつひとつの作りはきわめて単純なものです。

まず1500行までが、いろんな変数、スプライトのデータ、効果音のデータなどの初期化を行う部分で、この行以後が、本体のプログラムとなっています。

本体のプログラムは、キー入力、UFO 移動、エイリアンの移動、UFO にタマが当ったときの処理、エイリアンにタマが当ったときの処理、WARN の処理、ゲームオーバーの処理を行っています。ほとんどの処理が、UFO、エイリアン、照準の座標移動とそれに属す処理ですが、WARN 処理だけはタイマー割り込みを使って行っています。このタイマー割り込みというのは、**ON INTERVAL GOTO** で行う処理のことで、この命令で指定した時間になると、BASIC が今どの行の処理を行っていても、指定した行に処理を移すことができるものです。これによって正確な時間に WARN 処理を行うことが可能となったわけです。

ON～INTERVAL GOTO

さて、UFO やエイリアンそして照準ですが、これらは、すべてスプライトを使用しています。650行からのデータを見てもらえばわかると思いますが、UFO でも 3 つのデータがあります。これは、遠近感を出すためのもので、UFO が上にある場合は小さく、逆に下になるほど大きい表示を行い、距離感を出しているわけです。

SPRITE

なお、このプログラムは16Kバイト以上のシステムで使用可能で、ディスクシステムでも OK です。

リスト8(その1) バラン

```
10 '**********************************
20 '**                              **
30 '**           BARAN              **
40 '**                              **
50 '** program: by Nami aoki        **
60 '** debug and advice:            **
70 '**         by Haruka Takagi     **
80 '**                              **
90 '** copyright 1984 (C) NATS      **
100 '** copyright 1984 (C)          **
110 '**     Seibundo-Shinkousha     **
120 '**         publishing          **
130 '**                             **
140 '**********************************
150 MAXFILES=1
160 DEFINT A-Z:A=RND(-TIME)
170 DIM MS(13),MA(13),V(10),U(10),UX(10),UY(10),VX(10),
    VY(10)
180 SC=0:HS=0
190 SCREEN 2,2,0:COLOR 15,0,0:CLS
200 OPEN "GRP:" FOR OUTPUT AS #1
210 D$="O5L32T200V8"
220 PLAY D$,D$,D$
230 D1$="O5CDEFGABO6CDEFGABO7CDEFGAB":D2$="O5RCDEFGABO6
    CDEFGABO7CDEFGA":D3$="O5RRDEFGABO6CDEFGABO7CDEFG"
240 PLAY D1$,D2$,D3$
250 PRESET (0,30),1
```

```
260 PRINT #1,"  ******************************"
270 PLAY D1$,D2$,D3$
280 PRINT #1,"  *         B A R A N          *"
290 PRINT #1,"  *                            *"
300 PLAY D1$,D2$,D3$
310 PRINT #1,"  *   The Space fighter game   *"
320 PLAY D1$,D2$,D3$
330 PRINT #1,"  *    by N.Aoki & H.Takagi    *"
340 PRINT #1,"  *                            *"
350 PLAY D1$,D2$,D3$
360 PRINT #1,"  *     Copyright 1984 NATS    *"
370 PLAY D1$,D2$,D3$
380 PRINT #1,"  *     Copyright 1984 (C)     *"
390 PLAY D1$,D2$,D3$
400 PRINT #1,"  *     Seibundo-Shinkousha    *"
410 PLAY D1$,D2$,D3$
420 PRINT #1,"  ******************************"
430 IF PLAY(0) GOTO 430
440 SOUND 7,&HC7:SOUND 8,&H10:SOUND 9,&H10:SOUND 10,&H1
    0:SOUND 13,1
450 PRINT #1,:PRINT #1,
460 ' SPRITE PATTERN SET
470 '
480 RESTORE
490 FOR I=1 TO 11
500 A$=""
510 FOR J=1 TO 32
520 READB$:A$=A$+CHR$(VAL("&H"+B$))
530 NEXT J:SPRITE$(I)=A$
540 NEXT I
```

リスト8（その3）バラン

```
550 '
560 ' MUSIC DATA
570 '
580 FOR I=0 TO 13
590 READ A:MS(I)=A
600 NEXT
610 '
620 FOR I=0 TO 13
630 READ A:MA(I)=A
640 NEXT
650 '
660 ' DATA
670 '
680 '
690 ' DATA 1 UFO
700 DATA 0,0,0,0,0,1,9,F
710 DATA 9,0,0,0,0,0,0,0
720 DATA 0,0,0,0,0,80,90,F0
730 DATA 90,0,0,0,0,0,0,0
740 '
750 ' DATA 2 UFO
760 '
770 DATA 0,0,0,0,1,3,42,7F
780 DATA 43,5,0,0,0,0,0,0
790 DATA 0,0,0,0,80,C0,42,FE
800 DATA C2,A0,0,0,0,0,0,0
810 '
820 ' DATA 3 UFO
830 '
840 DATA 1,1,1,3,87,8F,FC,FF
850 DATA CF,87,9,10,20,0,0,0
860 DATA 80,80,80,C0,E1,F1,3F,FF
870 DATA F3,E1,90,8,4,0,0,0
```

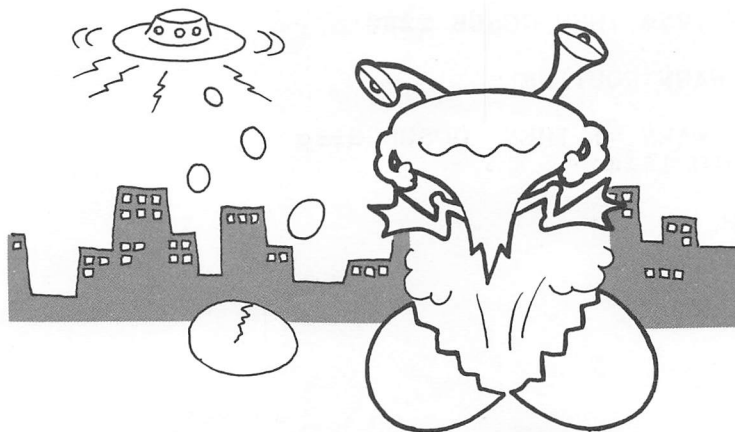
リスト8(その4) バラン

```
880 '
890 ' DATA 4 INVADER
900 '
910 DATA 0,3,3,7,1F,3F,3F,7F
920 DATA 7F,71,FF,11,20,40,20,10
930 DATA 0,C0,C0,E0,F8,FC,FC,FE
940 DATA FE,8E,FF,80,4,2,4,8
950 '
960 ' DATA 5 INVADER
970 '
980 DATA 0,3,3,7,1F,3F,3F,77
990 DATA 7B,7D,FE,89,11,21,40,80
1000 DATA 0,C0,C0,E0,F8,FC,FC,EE
1010 DATA DE,BE,FF,91,88,84,2,1
1020 '
1030 ' DATA 6 SHOJUN
1040 '
1050 DATA 1,1,1,1,1,1,1,FF
1060 DATA 1,1,1,1,1,1,1,1
1070 DATA 0,0,0,0,0,0,0,FF
1080 DATA 0,0,0,0,0,0,0,0
1090 '
1100 '
1110 ' DATA 7 TAMAGO
1120 DATA 0,0,0,7,F,1F,17,17
1130 DATA 17,1F,B,7,0,0,0,0
1140 DATA 0,0,0,80,C0,E0,E0,E0
1150 DATA E0,E0,CO,80,0,0,0,0
1160 '
1170 ' DATA 8 BAKUHATSU
1180 '
1190 DATA 7B,0,0,1F,3F,77,3F,17
1200 DATA ED,3F,5B,1,6F,10,18,A
1210 DATA 0,A0,EE,94,98,E8,90,EE
1220 DATA F0,DE,B4,80,B8,98,20,30
```

リスト8(その5) バラン

```
1230 '
1240 ' DATA 9  MISSILE  L
1250 '
1260 DATA 0,0,0,0,0,0,1,3
1270 DATA 4,0,0,0,0,0,0,0
1280 DATA 0,0,0,0,0,00,00,80
1290 DATA 40,0,0,0,0,0,0,0
1300 '
1310 ' DATA 10 MISSILE  M
1320 '
1330 DATA 0,0,0,0,0,0,01,03
1340 DATA 06,C,10,0,0,0,0,0
1350 DATA 0,0,0,0,00,00,00,80
1360 DATA C0,60,10,0,0,0,0,0
1370 '
1380 ' DATA 11 MISSILE  S
1390 '
1400 DATA 0,0,0,0,00,00,01,03
1410 DATA 07,0F,1C,38,60,80,0,0
1420 DATA 00,00,00,00,00,00,00,80
1430 DATA C0,E0,70,38,C,2,0,0
1440 '
1450 ' MUSIC DATA 1
1460 '
1470 DATA 0,0,0,0,0,0,1,7
1480 DATA 16,16,16,100,100,0
1490 '
1500 '
1510 ' MUSIC DATA 2
1520 DATA 0,0,100,3,24,0,16,1
1530 DATA 0,10,10,90,2,13
```

リスト8(その6) バラン

```
1540 '
1550 ON INTERVAL =200 GOSUB 2890
1560 ON STRIG  GOSUB 2610,2610,2610
1570 PRINT #1,"  JOY STICK (1) OR KEY BORD (2)"
1580 A$=INKEY$:IFA$="1" THEN JS=1 :GOTO 1600
1590 IF A$="2" THEN JS=0 ELSE GOTO 1580
1600 BEEP:BEEP:INTERVAL ON
1610 U(1)=1:UX(1)=RND(1)*32:UZ=1                :'UFO INI
     T.
1620 TI=0:UF=1:IV=1 :'TIMER INIT
1630 X=125:Y=80:CLS
1640 FOR I=0 TO 60:PSET(RND(1)*255,RND(1)*160),RND(1)*1
     4+2:NEXT
1650 FOR I=0 TO 80
1660 LINE (0+I,192)-(127,160),14
1670 LINE -(255-I,192),14
1680  NEXT
1690 FOR I=0 TO 40
1700 LINE (85+I,192)-(127,160),15
1710 LINE -(170-I,192),15
1720 NEXT:STRIG(JS) ON
1730 GOSUB 3240
```

リスト８（その７）バラン

```
1740 '
1750 ' MAIN ROUTINE
1760 '
1770 ' KEY IN !
1780 '
1790 ST=STICK(JS)
1800 ON ST GOTO 1820,1830,1840,1850,1860,1870,1880,1890
1810 GOTO 1900
1820 Y=Y-8:       GOTO 1900
1830 X=X+8:Y=Y-8:GOTO 1900
1840 X=X+8:       GOTO 1900
1850 X=X+8:Y=Y+8:GOTO 1900
1860 Y=Y+8:       GOTO 1900
1870 X=X-8:Y=Y+8:GOTO 1900
1880 X=X-8:       GOTO 1900
1890 X=X-8:Y=Y-8:GOTO 1900
1900 IF X<0 THEN X=240
1910 IF X>240 THEN X=0
1920 IF Y<0 THEN Y=160
1930 IF Y>160 THEN Y=0
1940 PUT SPRITE 0,(X,Y),15,6
1950 '
1960 ' UFO GET
1970 '
1980 IF UF>0 THEN GOSUB 2080
1990 '
2000 ' INVADER GET
2010 '
2020 IF IV>0 THEN GOSUB 2250
2030 '
2040 ' WARN ROUTINE
2050 '
2060 IF WARN =1 THEN  GOSUB 3180
2070 GOTO 1740
```

リスト8（その8） バラン

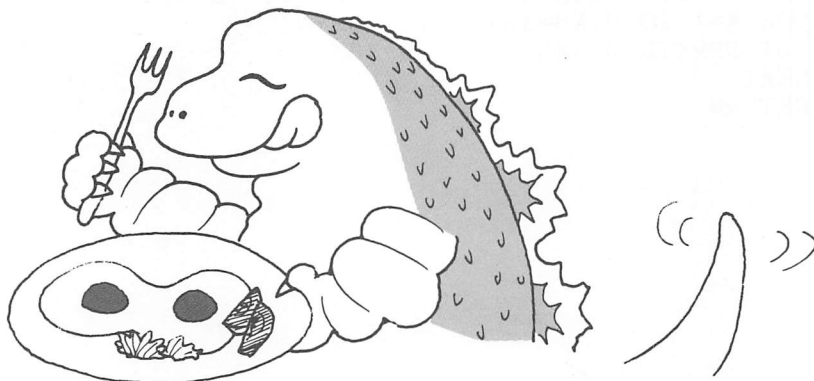
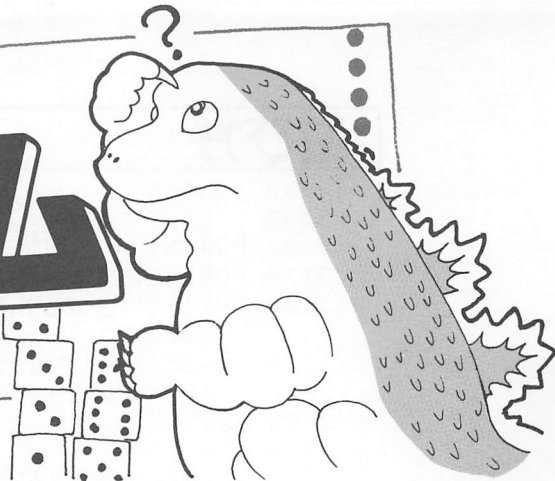
```
2080 '
2090 ' UFO ROUTINE
2100 '
2110 FOR I=1 TO UF
2120 UY(I)=UY(I)+RND(1)*U(I)*3*UZ
2130 UX(I)=UX(I)+TX+TX
2140 IF UY(I)>160 THEN UY(I)=150 : UZ=-1:BEEP
2150 IF UY(I)<0 THEN UY(I)=0: UZ=1
2160 IF UX(I)>240 THEN UX(I)=240:TX=-TX
2170 IF UX(I)<0 THEN UX(I)=0: TX=-TX
2180 IF U(I)=0 THEN 2230
2190 IF UY(I)<50 THEN U(I)=1
2200 IF UY(I)>50 THEN U(I)=2
2210 IF UY(I)>100 THEN U(I)=3
2220 PUT SPRITE 2+I, (UX(I),UY(I)),14,U(I)
2230 NEXT
2240 RETURN
2250 '
2260 ' INVADER ROUTINE
2270 '
2280 FOR I=1 TO UF
2290 IF U(I)=1 THEN 2330
2300 IF U(I)=3 THEN V(I)=1
2310 VX(I)=UX(I):VY(I)=UY(I)
2320 IF U(I)=2 THEN PUT SPRITE 14+I, (VX(I),VY(I)+8),10,
     7
2330 NEXT
2340 FOR I=0 TO IV
2350 IF V(I)=0 THEN 2420
2360 VX(I)=VX(I)+TA
2370 IF VX(I)>240 THEN VX(I)=240:TA=-TA
2380 IF VX(I)<0 THEN VX(I)=0: TA=-TA
2390 VY(I)=VY(I)+RND(1)*8
2400 IF VY(I) > 160 THEN 2470
2410 VZ=(VX(I)+VY(I)) MOD 2:PUT SPRITE 14+I, (VX(I),VY(I
     )),10,4+VZ
2420 NEXT
2430 RETURN
```

リスト8(その9) バラン

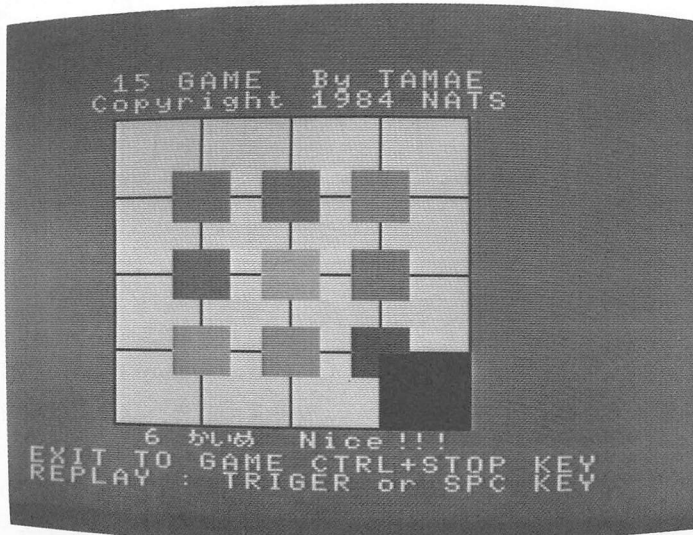
```
2440 '
2450 ' game over
2460 '
2470 FOR T=0 TO 13:SOUND T,MS(T):NEXT
2480 FOR TM=0 TO 100:Z=RND(1)*15+1:COLOR  ,Z,Z:NEXT
2490 COLOR ,0,0
2500 STRIG(JS) OFF :INTERVAL OFF:FOR I=0 TO10
2510 PRESET(0,80):PRINT #1,"        G A M E  O V E R  "
     :BEEP
2520 LINE (0,78)-(255,90),1,BF:BEEP
2530 NEXT
2540 PRESET (40,144):PRINT #1,"      TRY AGAIN (Y/N)"
2550 A$=INKEY$:IF A$="" THEN 2550
2560 IF A$="y" OR A$="Y" THEN SCREEN 2,2,0:SC=0:ERASE U
     X,UY,VX,VY,U,V:DIM UX(10),UY(10),VX(10),VY(10),U(10
     ),V(10):GOTO 460
2570 IF A$="n" OR A$="N" THEN END ELSE GOTO 2550
2580 GOTO 2550
2590 ' NICE ! hit ROUTINE
2600 '
2610 ' MISIILE TRIGGER ROUTINE
2620 '
2630 GOSUB 2990
2640 '
2650 ' UFO HIT !
2660 '
2670 FOR H=1 TO UF
2680 IF UX(H)< X+4+U(H) AND UX(H)>X-4-U(H) THEN AT=1 EL
     SE AT=0
2690 IF UY(H)< Y+4+U(H) AND UY(H)>Y-4-U(H) THEN AT=AT+1
2700 IF AT<2 THEN 2750
2710 PUT SPRITE 1,(X,Y),15,8
2720 FOR J=0 TO 13:SOUND J,MS(J):NEXT
2730 PUT SPRITE 1,(0,209):PUT SPRITE 2+H,(0,209):PUT SP
     RITE 14+H,(0,209)
2740 U(H)=1:UY(H)=0:UX(H)=RND(1)*255:SC=SC+10:GOSUB 324
     0
2750 NEXT H
```

```
2760 '
2770 '  INVADER HIT !
2780 '
2790 FOR H=1 TO IV
2800 IF VX(H)< X+8 AND VX(H)>X-8 THEN AT=1 ELSE AT=0
2810 IF VY(H)< Y+8 AND VY(H)>Y-8 THEN AT=AT+1
2820 IF AT<2 THEN 2870
2830 PUT SPRITE 1,(X,Y),15,8
2840 FOR J=0 TO 13:SOUND J,MS(J):NEXT
2850 PUT SPRITE 1,(0,209):PUT SPRITE 14+H,(0,209)
2860 VY(H)=0:VX(H)=RND(1)*255:SC=SC+50:V(H)=0:GOSUB 324
     0
2870 NEXT H
2880 RETURN
2890 ' TIMER INTERUPT
2900 '
2910 TI=TI+1
2920 TX=RND(1)*10-5
2930 TA=RND(1)*10-5
2940 IF TI MOD 20=0 THEN UF=UF+1:IV=IV+1:U(UF)=1:IF UF>
     6 THEN 3070 3100
2950 RETURN
2960 '
2970 ' MISILE & MISIILE SOUND
2980 '
2990 SOUND 7,254:SOUND 8,15
3000 FOR IN=1   TO255 STEP  3
3010 SOUND 0,IN:NEXT:SOUND 0,0
3020 XA=(X-120)/9:YA=(Y-160)/9:XB=120:YB=160
3030 FOR A=1 TO 9:XB=XB+XA:YB=YB+YA
3040 PUT SPRITE 0,(XB,YB),9,9+YB/53
3050 NEXT
3060 RETURN
```

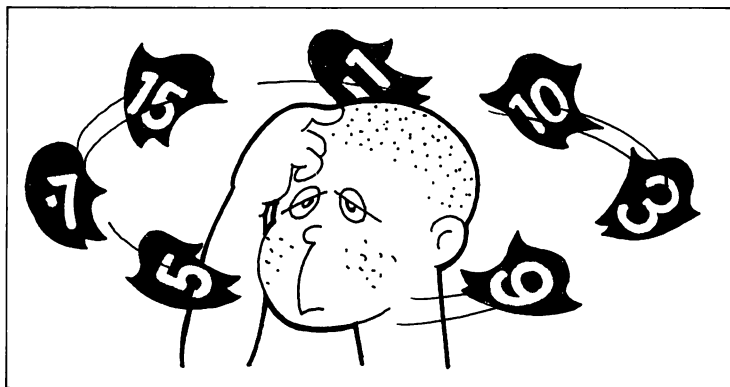
リスト8(その11) バラン

```
3070 '
3080 ' WARN !!
3090 '
3100 PRESET (100,8),8:PRINT#1,"WARN!!"
3110 FOR K=0 TO13:SOUND K,MA(K):NEXT
3120 UF=1:IV=6:FOR H=1 TO 6:V(H)=1:U(H)=0:PUT SPRITE 2+
     H,(0,209):VX(H)=RND(1)*240:NEXT
3130 SC=SC+1000:TI=0:U(1)=1:WARN=1
3140 'LINE (100,8)-(150,16),1,BF
3150 RETURN
3160 '
3170 ' WARN ROUTINE
3180 '
3190 FOR HJ=1 TO IV:IF V(HJ)=1 THEN 3210
3200 NEXT HJ: LINE (100,8)-(150,16),1,BF:IV=1:SC=SC+100
     0:WARN=0:GOTO 3230
3210 FOR K=0 TO 13:SOUND K,MA(K):NEXT
3220 IV=6
3230 RETURN
3240 IF SC>=HS THEN HS=SC
3250 SC$=RIGHT$("0000"+RIGHT$(STR$(SC),LEN(STR$(SC))-1)
     ,5):HS$=RIGHT$("0000"+RIGHT$(STR$(HS),LEN(STR$(HS))
     -1),5)
3260 LINE (0,0)-(255,8),0,BF:PRESET(0,0)
3270 PRINT #1,"  SCORE ";SC$;"    HI-SCORE ";HS$
3280 RETURN
```

15ゲームという名前は知らなくても、ほとんどの人がやったことのあるゲームだと思います。そうです。15個の数字や絵の書いてあるコマを動かし、ばらばらだった数字や絵をきちんとならべなおすゲームです。

コンピューターでやる場合は、数字ではつまりませんし、グラフィックだと、目がちらついてきて長時間遊ぶのはきついので、色をぬったタイルを使うことにしました。

また、最初の「かきまぜ」や完成までに何回コマを動かしたかなどは、すべてコンピューターがやってくれますので、人間様はゲームのみに集中することができます。

遊び方

まず**リスト9**を入力してください。次にRUN しますと、画面がクリアされてタイトルが出てジョイスティックを使うかどうか聞いてきますので、ジョイスティックを使う場合はy、本体のカーソルキーで遊ぶ場合はnを入力してください。

次に How many times shuffling ? (何回かきまぜますか) と聞いてきますので、かきまぜる回数を入力してください。だいたい初級で50回、中級で100回、上級で200回といったところでしょうか。あまりかきまぜると、にっちもさっちもいかなくなってしまいます。かきまぜる回数を入力すると、まず完成したところの絵が出て、かきまぜを始め、かきまぜが完了するとゲームがスタートします。後は、指でコマをかちゃかちゃと動かすのと同じ要領でカーソルキーでコマを動かす方向を指示してやればコマが1つずつ動きます。

プログラム

プログラムは大きく4つの部分から作られています。

まず、コマをかきまぜる部分(460行からのMIX UP)は、メモリー上にあるゲーム盤のコマをHow many times suffling ?で答えた回数かきまぜる部分で、かきまぜのキーワードは520行の**RND**関数による乱数によって行っています。

RND

次にコマを動かす部分(690行からのKOMA MOVE)ですが、ここでは、**STICK**関数を使ってキー入力またはジョイスティックを読

STICK

SWAP

み指定されたコマとコマのない部分を**SWAP**命令を使って交換し、表示を行います。これはたとえば上から下にコマを動かしたい場合、上のコマとコマのない部分の場所を交換すると、コマが上から下に動いたように見えることを利用しているわけです。

3番目はコマの表示です。この部分はマイコン内にあるコマの番号に対応した絵を書くもので、コマの番号は完成時左上から右へ0、1、2……となっていて、右下のコマのない部分が15番目になっています。ここでは、**ON GOTO**命令を使って絵を描くルーチンに分岐するようになっていますので、それぞれの番号に対応した飛び先に、自分の好きな絵を描くようにプログラムを変更すれば、あなただけの15ゲームができます（1090～1290行）。

ON GOTO

最後は、絵が完成したか？　というのを判定する部分です。ここは先ほど説明したコマの番号を見て判定しています。つまり、コマが0、1、2……14、15と並んでいれば、完成というわけですね。これは、820行からの GAME COMPLETE ? で行っています。

さてこのゲームは、時間よりもコマの移動回数の少なさを競うようになっていますが、これだけではつまらない、という人は時間もカウントし、両方の値によって評価を下すように変更するとおもしろいかもしれません。もちろん、時間は**TIME**変数を使うわけです。

TIME

なお、このプログラムは、16Kバイト以上のシステムで使用可能で、ディスクシステムでもOKです。

リスト9(その1) 15ゲーム

```
10 ' 15 game
20 ' by Tamae Tama
30 ' debug and advice by Haruka Takagi
40 ' copyright 1984 (C) NATS
50 ' copuright 1984 (C)
60 '     Seibundo-Shinkousha publishing
70 '
80 ' SYSTEM & HENSU INIT
90 '
100 DEFINT A-Z:KEY OFF
110 T$="15 GAME  By TAMAE"
120 C$="Copyright 1984 NATS"
130 SCREEN 1,0,0:CLS
140 PRINT "******************************";
150 PRINT "*          15 Game           *";
160 PRINT "*                            *";
170 PRINT "*   by T.Tama & H.Takagi     *";
180 PRINT "*  copyright 1984 (C) NATS   *";
190 PRINT "*     copyright 1984 (C)     *";
200 PRINT "*     Seibundo-Shinkousha    *";
210 PRINT "******************************";
220 LOCATE 5,15:INPUT "joystick(y/n)";A$
230 IF A$="Y" OR A$="y" THEN J=1
240 LOCATE 1,17:INPUT "How many times shuffling";SU
250 SCREEN 2:COLOR 15,4,4
260 OPEN "GRP:" FOR OUTPUT AS #1
270 '
280 ' KOMA INIT
290 '
300 CLS:FOR Y=0 TO 3
310 FOR X=0 TO 3
320 K(X,Y)=X+Y*4
330 NEXT
340 NEXT
350 PRESET(48,12):PRINT#1,T$
360 PRESET(40,20):PRINT#1,C$
370 BX=3:BY=3
```

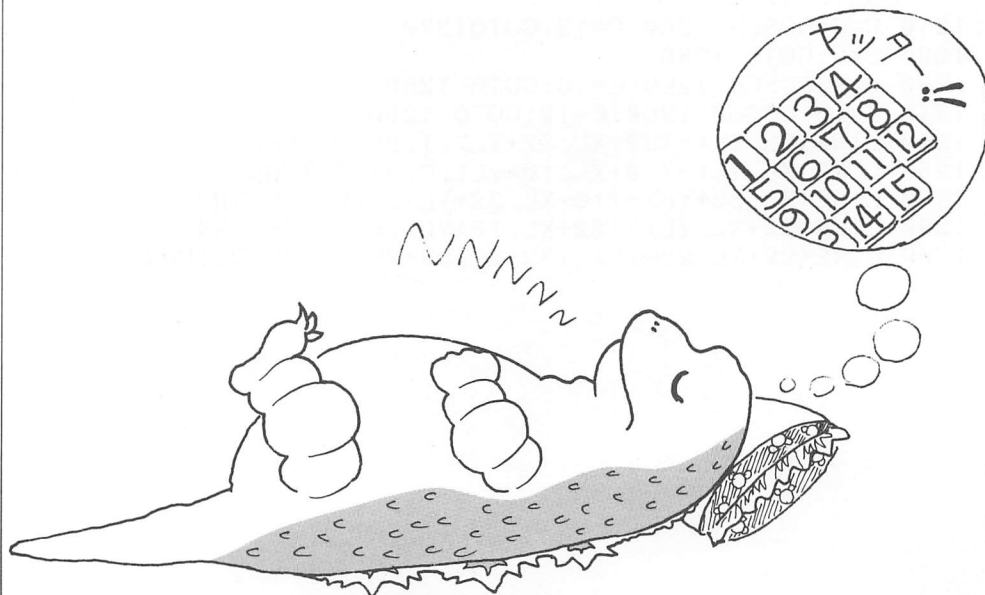
```
380 '
390 ' DRAW COMPLETE PICTUR
400 '
410 FOR Y=0 TO 3
420 FOR X=0 TO 3
430 GOSUB 1060
440 NEXT
450 NEXT
460 '
470 ' MIX UP
480 '
490 PSET(50,160),0:RN=RND(-TIME)
500 PRINT #1,"ﾀﾀﾞｲﾏ ｶｷﾏｾﾞﾃｲﾏｽ"
510 FOR I=0 TO SU
520 RN=RND(1)*4
530 IF RN=0 AND BY<>0 THEN SWAP K(BX,BY),K(BX,BY-1):BY=
    BY-1:GOTO 570
540 IF RN=1 AND BY<>3 THEN SWAP K(BX,BY),K(BX,BY+1):BY=
    BY+1:GOTO 570
550 IF RN=2 AND BX<>0 THEN SWAP K(BX,BY),K(BX-1,BY):BX=
    BX-1:GOTO 570
560 IF RN=3 AND BX<>3 THEN SWAP K(BX,BY),K(BX+1,BY):BX=
    BX+1 ELSE 520
570 NEXT
580 CLS
590 FOR Y=0 TO 3
600 FOR X=0 TO 3
610 GOSUB 1060
620 NEXT
630 NEXT
640 MOVE=1
650 PRESET(48,12):PRINT#1,T$
660 PRESET(40,20):PRINT#1,C$
670 PRESET(50,160)
680 PRINT #1,"1 かいめ"
```

リスト9（その3）15ゲーム

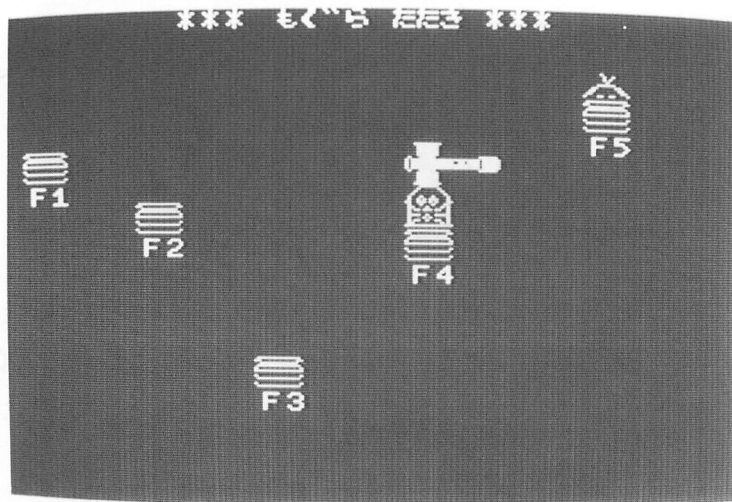
```
690 '
700 ' KOMA MOVE
710 '
720 I=STICK(J)
730 IF I=5 AND BY<>0 THEN BEEP:SWAP K(BX,BY),K(BX,BY-1)
    :BY=BY-1:X=BX:Y=BY:GOSUB1060:Y=Y+1:GOSUB 1060:GOTO
    780
740 IF I=1 AND BY<>3 THEN BEEP:SWAP K(BX,BY),K(BX,BY+1)
    :BY=BY+1:X=BX:Y=BY:GOSUB1060:Y=Y-1:GOSUB 1060:GOTO
    780
750 IF I=3 AND BX<>0 THEN BEEP:SWAP K(BX,BY),K(BX-1,BY)
    :BX=BX-1:X=BX:Y=BY:GOSUB1060:X=X+1:GOSUB 1060:GOTO
    780
760 IF I=7 AND BX<>3 THEN BEEP:SWAP K(BX,BY),K(BX+1,BY)
    :BX=BX+1:X=BX:Y=BY:GOSUB1060:X=X-1:GOSUB 1060:GOTO
    780
770 GOTO 720
780 MOVE=1+MOVE
790 LINE(152,168)-(52,160),0,BF
800 PRINT #1,MOVE;"かいめ";
```

リスト9（その4）15ゲーム

```
 810 '
 820 ' GAME COMPLETE ?
 830 '
 840 FOR Y=0 TO 3
 850 FOR X=0 TO 3
 860 IF 4*Y+X<>K(X,Y) THEN 720
 870 NEXT
 880 NEXT
 890 PRINT #1,"  Nice !!"
 900 PRESET(16,170)
 910 Z$="O5L32T200V15R8"
 920 D1$="CDEFGABAGFEDCABCD"
 930 D2$="R4CDEFGABAGFEDCABC"
 940 D3$="R4R4CDEFGABAGFEDCAB"
 950 PLAY Z$,Z$,Z$
 960 FOR TM=0 TO 1:PLAY D1$,D2$,D3$:NEXT
 970 IF PLAY(0) THEN 970
 980 PRINT #1,"EXIT TO GAME CTRL+STOP KEY
 990 PRESET(16,178)
1000 PRINT #1,"REPLAY : TRIGER or SPC KEY
1010 IF STRIG(J)=0 THEN 1010 ELSE 300
1020 GOTO 1020
```

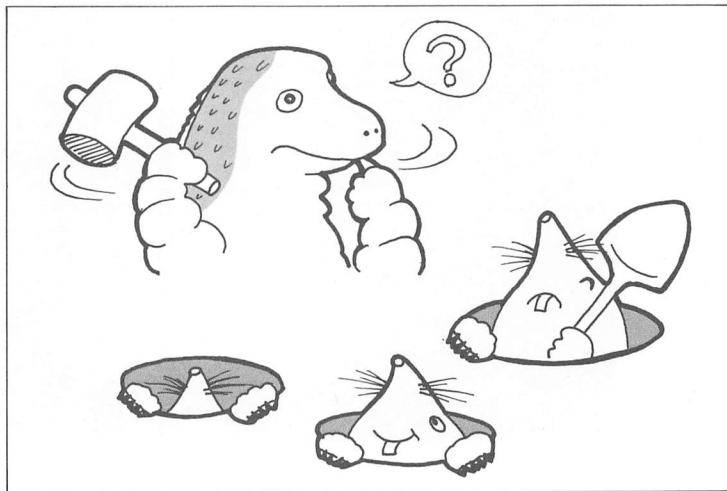
リスト9(その5) 15ゲーム

```
1030 '
1040 ' DRAW KOMA
1050 '
1060 XL=48+32*X:YL=30+32*Y
1070 LINE(XL,YL)-(32+XL,32+YL),15,BF
1080 LINE(XL,YL)-(32+XL,32+YL),1,B
1090 ON K(X,Y)+1 GOTO 1100,1110,1120,1130,1140,1150,116
     0,1170,1180,1190,1200,1210,1220,1230,1240,1250
1100 C=3:GOTO 1290
1110 C=3:GOSUB 1270:C=4:GOTO 1290
1120 C=4:GOSUB 1270:C=5:GOTO 1290
1130 C=5:GOTO 1270
1140 C=3:GOSUB 1280:C=6:GOTO 1290
1150 C=3:GOSUB 1260:C=6:GOSUB1270:C=4:GOSUB 1280:C=7:GO
     TO 1290
1160 C=4:GOSUB 1260:C=7:GOSUB1270:C=5:GOSUB 1280:C=8:GO
     TO 1290
1170 C=5:GOSUB 1260:C=8:GOTO 1270
1180 C=6:GOSUB 1280:C=9:GOTO 1290
1190 C=6:GOSUB 1260:C=9:GOSUB1270:C=7:GOSUB 1280:C=10:G
     OTO 1290
1200 C=7:GOSUB 1260:C=10:GOSUB1270:C=8:GOSUB 1280:C=12:
     GOTO 1290
1210 C=8:GOSUB 1260:C=12:GOTO1270
1220 C=9:GOTO 1280
1230 C=9:GOSUB 1260:C=10:GOTO 1280
1240 C=10:GOSUB 1260:C=12:GOTO 1280
1250 LINE(XL,YL)-(32+XL,32+YL),1,BF:RETURN
1260 LINE(XL,YL)-(10+XL,10+YL),C,BF:RETURN
1270 LINE(XL,22+YL)-(10+XL,32+YL),C,BF:RETURN
1280 LINE(22+XL,YL)-(32+XL,10+YL),C,BF:RETURN
1290 LINE(22+XL,22+YL)-(32+XL,32+YL),C,BF:RETURN
```

昔はゲームセンターでよくもぐらたたきをハンマーでひっぱたいている人を見かけましたが、最近はこの手のゲームはあまり見ませんね。

このゲームは、ゲームセンターのモグラたたきをシミュレートするものですが、あくまでも画面に出てくるモグラを、キーを押すことで「たたく」わけですから、MSX をとんかちでたたくようなことはやめておいたほうが良いと思います……やっぱり。

遊び方

まず**リスト10**を入力して、RUNします。タイトル画面が出て、難易度を聞いてきます。１を入力するとモグラの動きが一番速く、３が一番遅くなります。また３でもまだ速い！　という人は３以上を入れても受け付けるように作ってあります。とは言っても、しょせんBASICですので、それほど速くはないはずです。難易度を入力すると次はゲームがスタートします。

画面には５つのモグラの穴が表示され、そのそばにＦ１、Ｆ２などと書かれておりますが、これがファンクションキーの番号に対応しており、対応するファンクションキーを押せばその穴をたたくようになっています。

モグラは、ちょっと顔を出すときと、完全に出したとき、そして消えかけてる時の３通りがあり、ハンマーでたたいて有効となるのは、顔を出した状態から、完全に出すときまでのほんの少しの時間です。この間にモグラをたたきますと、モグラはキュンと鳴いて赤くなります。モグラが100回顔を出すと、ゲームオーバーとなり、たたいた回数と当った回数などの結果から、コメントが出ます。なお、あまりたたきすぎますと注意が出ますが、愛機のキーボードのためにもたたきすぎに注意しましょう。

プログラム

READ B$
SPRITE $

　このゲームは、穴、モグラ、ハンマーすべてにスプライトを使用しています。スプライトのデータは880～1220行に入っており、このデータを変更することにより、オリジナルのもぐらたたきを作ることができます。

ON KEY GOSUB

　モグラ、ハンマーともにスプライトを使用することで高速化ができたわけですが、実はもう一つ重要な命令、**ON KEY GOSUB** 命令を使っています。この命令は、今プログラムがどこの行を実行していようとも、ファンクションキーが押されると、この命令で指定した行に制御を移すことができるもので、いつ押されるかわからないキーに対して高速の処理を行うことが可能となります。

ON INTEVAL GOSUB
ON SPRITE GOSUB
ON STRIG GOSUB

　一般にこのような処理方式を「割り込み」と呼んでいますが、MSX-BASIC では、ファンクションキーによる割り込みのほか、一定時間による割り込み (**ON INTEVAL GOSUB**) スプライトの衝突による割り込み (**ON SPRITE GOSUB**) やジョイスティックによる割り込み (**ON STRIG GOSUB**) などゲームを作るのに便利な命令がたくさん用意されています。

　このプログラムでは、ファンクションキーが押されると、そのキーに対応する各処理に分岐し、そのときのモグラの状態を見てモグラをうまくたたいたかどうかの判断をしています。

　逆にこの方法で困る点は、ファンクションキーが押されっぱなしですと、この割り込み処理から抜け出ることができなくなり、動作がストップしてしまうことです。この点については、BASIC の特性ですのでどうしようもありません。

　なお、このプログラムは16Kバイト以上のシステムならどんなものでも動作します。

リスト10(その1) もぐらたたき

```
 10 ' もく゛らたたき ケ゛ーム
 20 ' by Nami Aoki
 30 ' debug and advice by Haruka Takagi
 40 ' copyright 1984 (C) NATS
 50 ' copyright 1984 (C)
 60 '    Seibundo-Shinkousha publishing
 70 SCREEN 0:CLS
 80 PRINT "****************************************";
 90 PRINT "*             モク゛ラ たたき ケ゛ーム            *";
100 PRINT "*           by N.Aoki & H.Takagi         *";
110 PRINT "*                                        *";
120 PRINT "*           copyright 1984 NATS          *";
130 PRINT "*           copyright 1984 (C)           *";
140 PRINT "*     Seibundo-Shinkousha publishing    *";
150 PRINT "****************************************";
160 PRINT :PRINT
170 INPUT "      ナンイト゛ハ ? (1(HARD)-3(EASY))";MU:SC=0:MS=
    0:FF=0
180 SCREEN 2,2,0:COLOR 15,1,7
190 ON KEY GOSUB 630,680,730,780,830
200 KEY(1) ON:KEY(2) ON:KEY (3) ON
210 KEY(4) ON:KEY(5) ON
220 OPEN "GRP:" FOR OUTPUT AS #1
230 FOR I=1 TO 5:A$=""
240 FOR J=0 TO 31
250 READ B$:A$=A$+CHR$(VAL("&H"+B$))
260 NEXT J
270 SPRITE$(I)=A$
280 NEXT I
290 X(1)=10:Y(1)=34:X(2)=50:Y(2)=54
300 X(3)=90:Y(3)=114:X(4)=140:Y(4)=64
310 X(5)=200:Y(5)=14
```

```
320 '
330 ' SCREEN SET
340 '
350 CLS:PRESET (0,0):PRINT #1,"         *** もぐ゛ら たたき ***
"
360 PUT SPRITE 20,(10,50),3,5
370 PRESET STEP(3,16):PRINT #1,"F1"
380 PUT SPRITE 21,(50,70),3,5
390 PRESET STEP(3,16):PRINT #1,"F2"
400 PUT SPRITE 22,(90,130),3,5
410 PRESET STEP(3,16):PRINT #1,"F3"
420 PUT SPRITE 23,(140,80),3,5
430 PRESET STEP(3,16):PRINT #1,"F4"
440 PUT SPRITE 24,(200,30),3,5
450 PRESET STEP(3,16):PRINT #1,"F5"
460 '
470 ' MAIN ROUTINE
480 '
490 FOR TI=0 TO MU*3:R=INT(RND(1)*5+1):NEXT
500 IF MU=3 THEN 610
510 IF MA(R)=1 THEN 530
520 MA(R)=1:MI(R)=1
530 FOR I=1 TO 5
540 IF MI(I)=0 THEN 590
550 IF MI(I)=1 THEN MI(I)=2 :PUT SPRITE 10+I,(X(I),Y(I)
    ),13,2:GOTO 590
560 IF MI(I)=2 THEN MI(I)=3 :PUT SPRITE 10+I,(X(I),Y(I)
    ),13,1:BEEP:MS=MS+1:IF MS > 99  THEN 1470 ELSE 590
570 IF MI(I)=3 THEN MI(I)=4 :PUT SPRITE 10+I,(X(I),Y(I)
    ),13,2:GOTO 590
580 MI(I)=0:MA(I)=0:PUT SPRITE 10+I,(X(I),209)
590 NEXT I
600 GOTO 490
610 FOR I=0 TO 5:IF MA(I)<>0 THEN 480
620 NEXT:MI(R)=1:GOTO 530
```

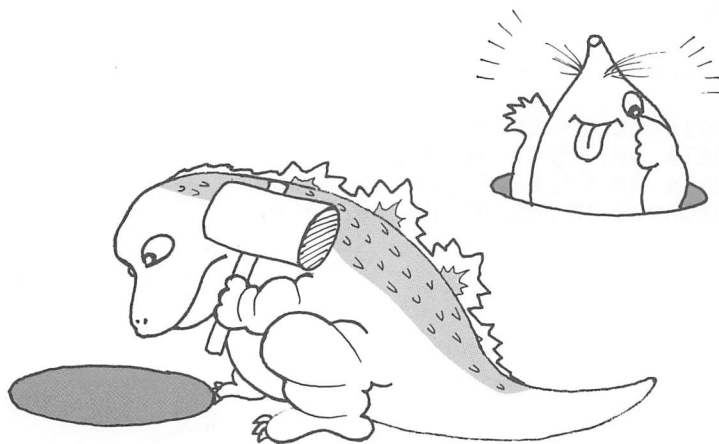
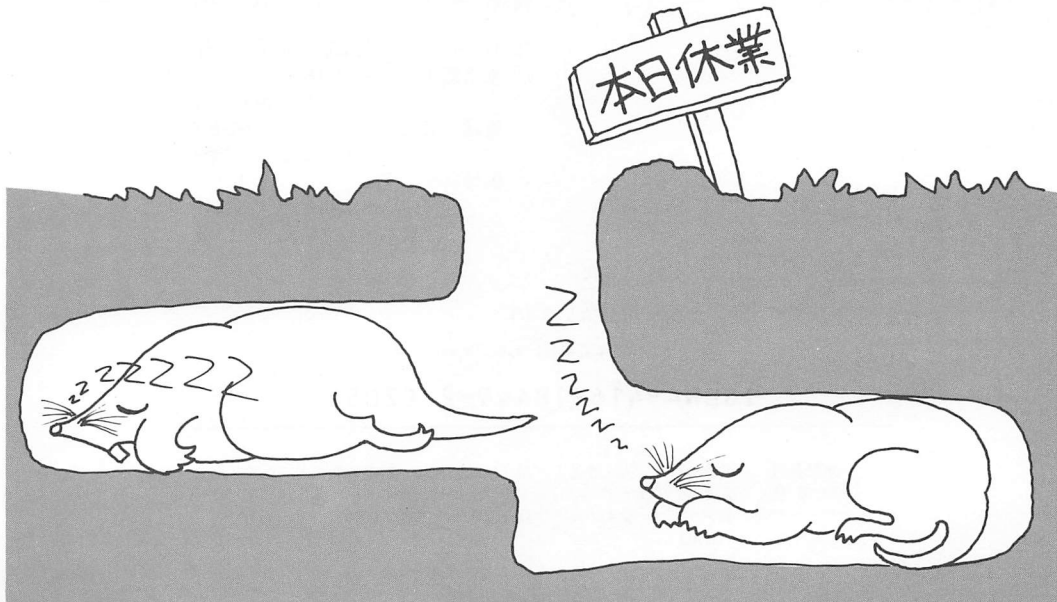
リスト10(その3) もぐらたたき

```
630 '
640 ' F1 HIT
650 FU=1:GOSUB 1260
660 IF MI(1)=3 THEN PUT SPRITE 11,(X(1),Y(1)),9,1:GOSUB
     1390
670 GOTO 1460
680 '
690 ' F2 HIT
700 FU=2:GOSUB 1260
710 IF MI(2)=3 THEN PUT SPRITE 12,(X(2),Y(2)),9,1:GOSUB
     1390
720 GOTO 1460
730 '
740 ' F3 HIT
750 FU=3:GOSUB 1260
760 IF MI(3)=3 THEN PUT SPRITE 13,(X(3),Y(3)),9,1:GOSUB
     1390
770 GOTO 1460
780 '
790 ' F4 HIT
800 FU=4:GOSUB 1260
810 IF MI(4)=3 THEN PUT SPRITE 14,(X(4),Y(4)),9,1:GOSUB
     1390
820 GOTO 1460
830 '
840 ' F5 HIT
850 FU=5:GOSUB 1260
860 IF MI(5)=3 THEN PUT SPRITE 15,(X(5),Y(5)),9,1:GOSUB
     1390
870 GOTO 1460
```

```
880 '
890 ' MOGURA
900 '
910 DATA 07,08,10,20,44,4A,CE,44
920 DATA 61,5B,40,79,42,59,60,FF
930 DATA E0,10,08,04,42,E2,A3,46
940 DATA 0A,B2,02,3E,82,32,0A,FF
950 '
960 ' MOGURA HALF
970 '
980 DATA 00,00,00,00,00,00,00,02
990 DATA 01,01,00,07,18,30,60,86
1000 DATA 00,00,00,00,00,00,00,00
1010 DATA 20,40,80,E0,18,0C,06,61
1020 '
1030 ' HAMMAR 1
1040 '
1050 DATA 1F,0F,0F,0F,0F,47,BF,BF
1060 DATA BF,BF,47,0F,0F,0F,0F,1F
1070 DATA F0,E0,E0,E0,E0,C0,FF,FF
1080 DATA FF,FF,C0,E0,E0,E0,E0,F0
1090 '
1100 ' HAMMAR 2
1110 '
1120 DATA 00,00,00,00,00,00,FF,37
1130 DATA FF,FF,00,00,00,00,00,00
1140 DATA 00,00,00,00,00,7E,BF,BF
1150 DATA BF,BF,7E,00,00,00,00,00
1160 '
1170 ' ANA
1180 '
1190 DATA 00,7F,C0,3F,7F,80,FF,00
1200 DATA 80,7F,00,80,7F,00,00,00
1210 DATA 00,FE,03,FC,FE,01,FF,00
1220 DATA 01,FE,00,01,FE,00,00,00
```

リスト10(その5) もぐらたたき

```
1230 '
1240 '
1250 '
1260 '
1270 ' HIT HAMMAR PRINT
1280 '
1290 FOR K=1TO5:KEY(K) OFF:NEXT
1300 PUT SPRITE 0,(X(FU),Y(FU)-16),11,3
1310 PUT SPRITE 1,(X(FU)+16,Y(FU)-16),11,4
1320 PLAY "L64O7FG"
1330 IF PLAY(0) THEN 1330
1340 PUT SPRITE 0,(0,209)
1350 PUT SPRITE 1,(0,209)
1360 FF=FF+1
1370 PLAY "L64O7DE"
1380 RETURN
1390 '
1400 ' HIT SOUND
1410 '
1420 FOR SO=0 TO 255
1430 SOUND 0,SO
1440 NEXT :SC=SC+1:RETURN
1450 MI(FU)=0:MA(FU)=0
1460 FOR K=1 TO 5 :KEY(K) ON:NEXT :RETURN
```

リスト10(その6) もぐらたたき

```
1470 ' GAME OVER
1480 '
1490 FOR K=1 TO 5:KEY(K) OFF:NEXT:SCREEN 1:LOCATE 10,5:
     PRINT"♥GAME OVER♥"
1500 LOCATE 8,5:PRINT" モク゛ラ  :  100 ヒ゜キ"
1510 LOCATE 8,7:PRINT" ヒット   :";SC;"カイ"
1520 LOCATE 8,9:PRINT" ハンマー :";FF;"カイ"
1530 QA=SC/FF:LOCATE 3,12
1540 IF QA<=.5 THEN PRINT "まあ まあ の せいせき て゛すね          ":G
     OTO 1570
1550 IF QA<=.7 THEN PRINT "なか なか すは゛らしい て゛すね          ":G
     OTO 1570
1560                    PRINT "ちょうし゛んてきな はんしゃしんけいて゛す "
1570 IF FF>100 THEN LOCATE 3,15 :PRINT "あまり たたきすき゛ないように
     してくた゛さい!!!"
1580 LOCATE 8,19:PRINT" TRY AGAIN ?";
1590 A$=INKEY$:IF A$="Y" OR A$="y" THEN RUN
1600 IF A$="N" THEN SCREEN 0:END
1610 IF PLAY(2) THEN 1590
1620 PLAY"T180O6L16ER32ER32GR32ER16 BR32ER32GR32ER","T2
     00V5O4L4CRO3GR","T200V5O4L4GRO4DR"
1630 GOTO 1590
```

検印省略　　NDC 548

MSXソフト集 Part 1

昭和60年5月10日　発　行

定価　850円

編　者　N A T S

発行者　小 川 茂 男

発行所　株式会社 誠文堂新光社
東京都千代田区神田錦町1－5－5
郵便番号 101
電話　東京 (292) 1 2 1 1 (編集)
(292) 1 2 2 1 (営業)
振替口座　東京　7－6294
印刷・広研印刷株式会社
製本・岡 嶋 製 本 工 業

ISBN4-416-18449-2 C2055

本書のプログラムで、お気づきの点がありましたら、
手紙にて当社初ラ編集部MSX係宛に御一報下さい。